IMPLEMENTATION OF A RELATIONAL DATA BASE MANAGEMENT SYSTEM AS AN EXTENSION TO FORTRAN

A Thesis Submitted
In Partial Fulfilment of the Requirements
for the Degree of
MASTER OF TECHNOLOGY

by

LT COL RAJINDER KUMAR BAGGA

to the

COMPUTER SCIENCE PROGRAMME
INDIAN INSTITUTE OF TECHNOLOGY KANPUR
AUGUST 1977

CSP-1977-M-BAG-IMP

To

Veena

Who endured my infatuation,
with the Unpredictable TDC!

CERTIFICATE

This is to certify that the thesis entitled "IMPLEMENTATION OF A RELATIONAL DATA BASE MANAGEMENT SYSTEM AS AN EXTENSION TO FORTRAN" by Lt. Col. Rajinder Kumar Bagga is a record of work carried out under my supervision and has not been submitted elsewhere for a degree.

[illegible]

Kanpur
August 1, 1977

V. Rajaraman
Professor of Computer Science & Electrical Engineering
and
Head, Computer Centre
INDIAN INSTITUTE OF TECHNOLOGY, KANPUR

ACKNOWLEDGEMENT

I wish to express my gratitude to Professor V. Rajaraman for his inspiring guidance throughout the present work. He has been very kind in sparing his valuable time for regular fruitful discussions.

To the DBMS group (Ajoy, Deepak and Ravi) headed by Professor R. Sankar, I am thankful for many frank and stimulating discussions, especially during the formulating stages of the thesis.

I owe thanks to Sri S.P. Khatri for assisting me in KDM interface and Sri M.V.S. Rao for keeping my hopes alive on TDC-316. Lt. Gian Prakash, IN, deserves special mention for his artistic touches in preparing the flowcharts.

Finally I must thank Sri H.K. Nathani for all the typing work.

RkBagga

Kanpur
August 1977

Lt. Col.
R.K. Bagga

ABSTRACT

An experimental Relational Data Base Management System has been implemented on TDC-316 at Indian Institute of Technology, Kanpur. The system provides most of the facilities required for Data Model Definition, Data Sub Language and Data Manipulation Language. The system has been extensively tested with live data from two laboratories generating 11 relations, and 28 distinct fields of different types and length. The results obtained have been encouraging.

For typical applications of DBMS in monitoring progress of various jobs/projects, a Relational approach has been preferred to Hierarchical or Network Approach for its simplicity and ease of implementation. Though inefficient in Data Processing environment TDC FORTRAN has been used in preference to BAL-316, because of its popularity and available system routines like SORT, MERGE. The system software, though experimental can be implemented in its present form in any organisation having TDC-316.

TABLE OF CONTENTS

CHAPTER I

INTRODUCTION

1.1 Historical Background

In early days of Electronic Data Processing (EDP), machine-readable collections of data were stored on external media such as cards or tapes. Beginning in the late fifties and early sixties data banks were being stored on-line, using direct-access storage devices (DASD) such as disks. During late sixties and early seventies, the idea of an integrated Data Base Management Systems (DBMS) was developed. This concept allowed several applications to share a common bank of data, maintained and protected by a central system. In an integrated data-base environment, the system provides each application program with its own view of the common data, implements various operations for retrieval and updating of data, and resolves interference between concurrent users.

1.2 Definitions

An elaborate definition of DBMS has been given by J. Martin as "a collection of interrelated data stored together with controlled redundancy to serve one or more applications in an optimal fashion; the data are stored so that they are independent of programs which use the data; a common and controlled approach is used in adding new data and modifying and retrieving existing data within the database" (1). Engles has given a working definition of data-base "A data-base is a collection of stored operational data used by the application systems of some particular enterprise." (2). Enterprise has been used as generic term for any large scale commercial, scientific, technical or other operation.

1.3 Objectives of DBMS

The major objectives of a centralised DBMS can briefly be enumerated as:

1-3.1 Data Availability: Data Availability involves sharing of stored data. Not only all the files of existing applications are integrated, but also new applications may be developed to operate against the existing data-base.

1-3.2 Data Quality: Data Quality means maintenance of quality of data and the integrity of the system. The data may have poor quality because it was:

(a) never any good;
(b) altered by human error;
(c) altered by program with a bug;
(d) altered by machine error; or
(e) destroyed by major catastrophe.
(e.g., a mechanical failure of disk etc.)

The maintenance of quality involves the detection of error, determination of how the error occurred and correction of the errorneous data, with preventivel action to avoid repetition of the error. The integrity of the data can be maintained by input-validation techniques in original data definition, logging of data base changes, periodic snapshots of the entire machine status and total or incremental data-base dumping. (3)

1-3.3 Data Independence: That is the immunity of applications program to change in storage structure and access strategy. This is the most important objective which encompasses both physical and logical data independence

(a) Physical data independence. A system is data independent, if the program or adhoc requests are relatively independent of the storage or access methods.

(b) Logical data independence: The ability to make logical change to the data base without significantly affecting the programswhich access it.

Data independence involves two aspects; first the capability of DBMS to support various (system or user) views of the database, and second, the capability of DBMS to allow modifications of these views without adversely affecting the integrity of the existing applications. This definition of data independence is perhaphs too broad. It suggests that substantial logical changes could be made without creating a need to change the program - a difficult, if not impossible task. (3).

1-3.4 Reduction of Data Redundancy: The amount of redundancy in the stored data should be reduced and possibly eliminated. The current systems (each application having its own private files) often lead to considerable redundancy in stored data with resultant waste in storage space. As a corollary, the problems of inconsistancy in the stored data can be avoided to a large extent.

1-3.5 Privacy and Security: The need to protect the data base from inadvertent access or unathorised discolosure; is achieved through some security mechanism. Security audits (audit trail) are achieved by logging access by people and programs to any secure information. These mechanisms allow Data Base Administrator (DBA) to determine who has been accessing what data, under what conditions, thereby monitoring possible leakage and preventing any threat to privacy.

1-3.6 Management Control: The establishment of the data administration functions and design of effective database. With central control of data base, the DBA can ensure that installation and industry standards are

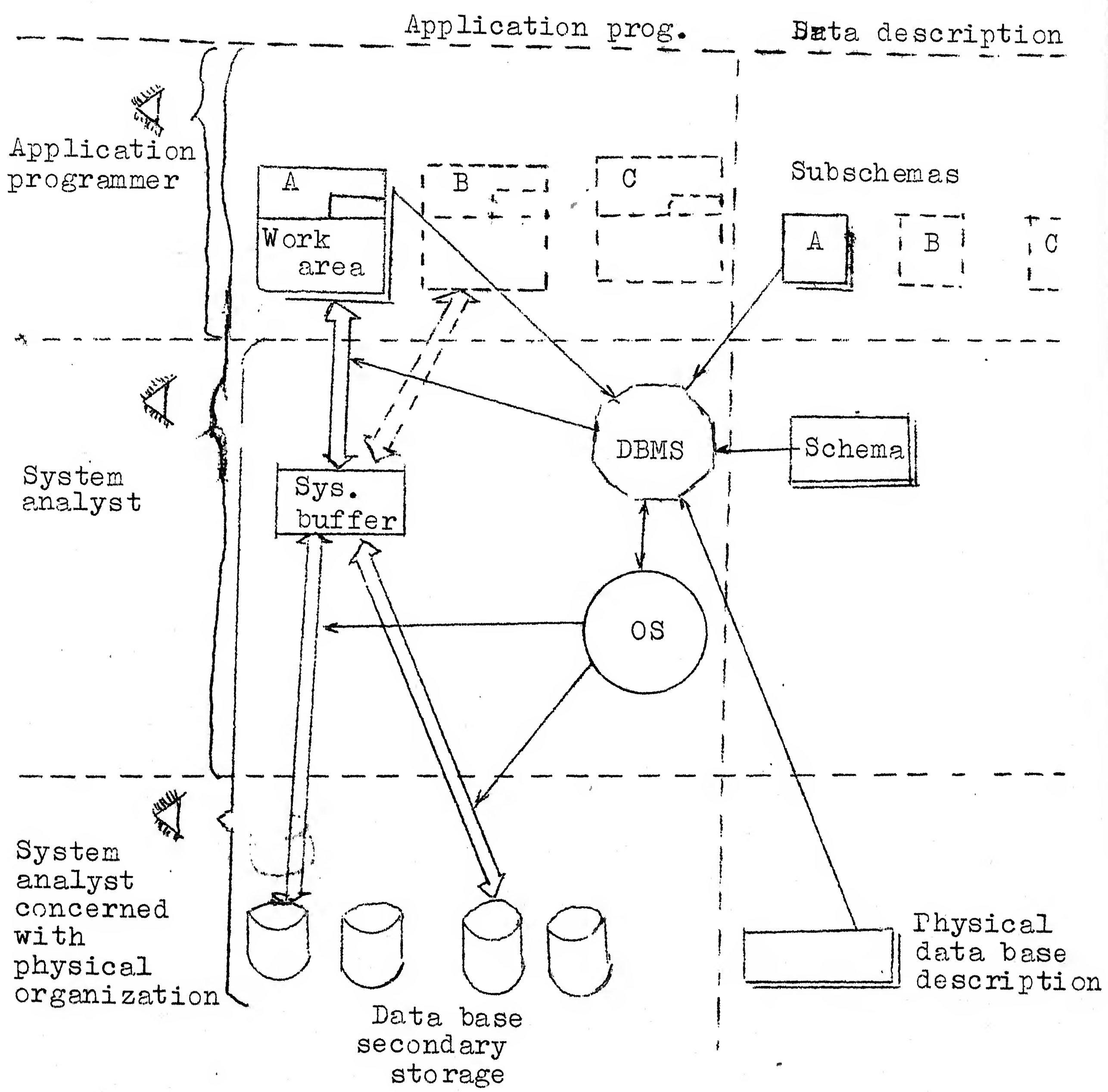

Application prog.
Data description
Application programmer
A
B
C
Work area
Subschemas
A
B
C
System analyst
DBMS
Schema
Sys. buffer
OS
System analyst concerned with physical organization
Physical data base description
Data base secondary storage

Figure 1.1

followed in the representation of the data. Using his knowledge of the requirements of the enterprise as opposed to the needs of any individual user – the DBA can structure database system to provide an overall service which is best for the enterprise. Thus the objectives of DBMS design involves trade offs, because users may have quite incompatible requirements

1.4 Generalised Architecture for a DBMS

The major components of the architecture for a typical DBMS are shown in Fig. 1.1[4]. These are:

1-4.1 User View: An application programmer's or remote terminal user's view having any degree of sophistication. Each user has his work area and uses a language which may be an independent Data Sub Language (DSL) or a data sublanguage embedded in host language. DSL in a way defines a subschema for individual user.

1-4.2 Analysts Physical Storage View: The data base is physically recorded on secondary storage using physical data description software.

1-4.3 System Analyst/DBA View: In between the data base and the users is interposed the Data Model/data sub model. The model is simply a view of the data base as it is viewed by the users. The mapping of the data model into physical storage is provided by DBMS software and operating system. Schema provides the Data Definitions for the entire data base. DBA's responsibilities include the following:

(a) Deciding information contents of data base.

(b) Deciding the storage structure and access strategy.

(c) Liaison with users

(d) Defining authorization checks and validation procedure.

(e) Defining a strategy for backup and recovery.

(f) Monitoring performance and responding to changes in requirement (5

1.5 Overview of Thesis

1-5.1 The importance of providing accurate information in the shortest possible time to heads of organization whether Government or private - cannot be over emphasized. The present work is aimed at providing one such facility at any central location where all data is centrally stored - and different types of querries are required to be answered. For example Defence Research and Development Organization (DRDO) has a network of laboratories for undertaking different types of projects. The progress of each of the projects is required to be monitored at DRDO Headquarters located at Delhi. This monitoring can be done with the help of the software developed.

1-5.2 At present there are the undermentioned three approaches for designing a Data Model for various applications:

(a) Hierarchical approach
(b) Network or DBTG Approach
(c) Relational Approach.

1-5.3 After a brief introduction on DBMS in Chapter 1, the comparative evaluation of the three approaches has been done in Chapter II. For the type of application envisaged it is felt that Relational approach may be more suitable. The design details of the relational approach with special emphasis on normalization and system architecture have been presented in Chapter III. Chapter IV is devoted to the implementation details of Data Model Definition (DMD) Analyser, Data Sub Language (DSL) and Data Manipulation routines. Within the time and resources available system performanc of the Relational Data Base Management system hasbeen evaluated. The report is given in Chapter V. In the concluding chapter the scope for future work on the subject has been recommended.

1.6 Choice of Computer System for Implementation

1-6.1 An indigeneously designed computer system TDC-316 (a 16 bit, third generation minicomputer) with the following configuration has been installed at Indian Institute of Technology, Kanpur:

a. TDC-316 processor
b. 28K word of core memory
c. ASR 33 teletype device (KBD & TTY)
d. Low speed reader (LSR)
e. High speed reader (HSR)
f. High speed punch (HSP)
g. 600 CPM card reader (CR)
h. 300 LPM Line Printer (LP)
j. 7.8M Byte disk Unit.
k. Low speed Punch (LSP)

1-6.2 The choice of the system for DBMS implementation was made primarily because of the ready availability of the computer time for software development on TDC-316. Certain hardware features like stack, bus for I/O, could be made use of, provided suitable language is chosen for implementation.

It was further felt that the development of DBMS package for an indegeneous computer, would benefit at large number of Indian users in various Government departments, business houses, universities and laboratories. Nearly 35 TDC 316 have already been marketed in India to the various agencies.

1.7 Language for Implementation

1-7.1 Since BRASS was not fully implemented, the choice of language for implementation was strictly between BAL-316 and FORTRAN. COBOL, the most commonly used language for Data Processing in the world, has yet to be implemented on TDC-316. The compiler is still under development at the Electronics Corporation of India, Hyderabad. ALGOL compiler is also not ready. In addition to the above mentioned reasons, it was decided to use TDC FORTRAN for partial implementation of Relational Data Base Management System, because:

(a) FORTRAN is a widely accepted machine independent language.

(b) With subroutine structure available, there would be no need to rewrite the compiler for extension of FORTRAN for DBMS.

(c) A number of programs written in FORTRAN to provide various utilities like SORT, MERGE could be readily used.

(d) The software developed in FORTRAN could be transported to any other machine having FORTRAN compiler with ease.

(e) Time available for implementation of complete system was approximately 6-man months. In such a short time it would have been very difficult to use BAL-316.

(f) Being a High Level Language with the following special features programming would be easier:

(i) O,T and single quote FORMATS
(ii) Adjustable Execution Time FORMATS using N
(iii) Width free FORMATS
(iv) Conditional compilation facility
(v) Complete mixed mode arithmetic etc.

1-7.2 However one of the major constraints in using FORTRAN would be the limited run time memory of 16K words including users work space. Giving 2-4K word to user, entire software, incorporating Data Definition Language, Search routines, Data manipulation features of DSL, should be completed in 12K words. In a similiar type of software developed for partial implementation of CODASYL DBTG report on CDC 6500 computer, it has taken 30K memory (6).

CHAPTER II

EVALUATION OF APPROACHES

2.1 General Terminology

The evolving field of data model approaches is often hotly debated. Proponents of each model point out advantages, but so far there is no consensus as to the best version. A DBMS generally supports one of the data models i.e., hierarchy, network or relational. Since each model uses a different terminology, Table1 attempts to equate the various terms with the concepts that have been developed (3).

Concept	Relational	Network	Hierarchical
Item	Role name/domain	Data item type	Item/field
Item value	component	data item occurence	value
Group	not allowed	group	group
Entity (type)	relation	record type	entry/segment type
Entity instance	tuple	record occurrence	entry/segment occurrence
Data administrative view	Data model (DM)	Logical structure	logical structure
Definition of DBA view	DM definition	Schema	Schema
User view	Data sub model	subschema	subschema
Entry point	Primary key	singular sets CALC records	root group root segment
Single unique identifier	candidate key	key	sequencer (unique)

Table 2.1 Comparative Terminology

2.2 Hierarchical Approach

2-2.1 For historical reasons this approach is very popular; it is used in many existing data base systems including IBM's Information Management System (IMS). It has its origin in the storage structure which were prevalent when most data processing was performed with purely sequential media, and there was only a minimal distinction between the data model and the storage structure. Figure 2.1 presents a hierarchical model of two laboratories with their projects, where LAB's are superior to projects.

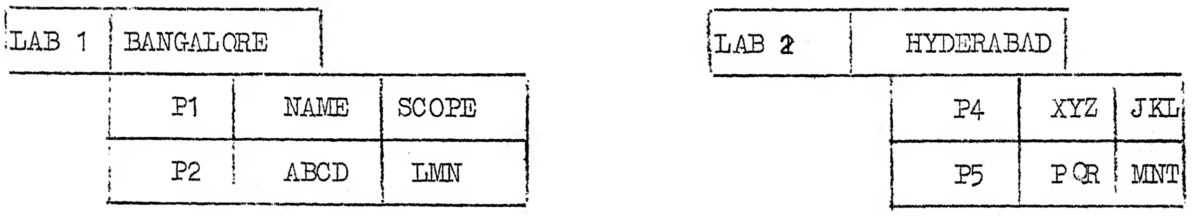


Fig. 2-1 Lab-and-Project data model in hierarchical form.

2-2.2 Each occurrence consists of one lab segment occurrence together with one project segment for each of the projects. The unit of access - i.e., the smallest amount of data which may be transferred by one DSL statement is normally one segment occurrence. In hierarchical model - a hierarchy of entities is involved - a superior entity and one or more inferior entities, each of which may participate as superior entitites at a third level. A hiearchy represents a 'tree', 'bush' or 'fan-out' of entities all related by family tree-like relationship (with no sons shared by different fathers). The top most level of hiearchy is termed the entity or root.

2-2.3 The major advantage of the hierarchical approach is that it obviously provides a very natural way of modelling a hierarchical structure from real world which generally are one to many type. However, difficulties arise when we try to operate on many to many relationship. By the way of illustration, let us consider two sample queries on Lab-Project model and DSL algorithm required to answer these using hierarchical approach (Table 2.2).

Q1. Find Project # for the Project under taken by LAB2?	Q2. Find LAB # having Project # P2?
Get unique with LAB # = LAB 2 Next: Get next part for this LAB Project found? If not EXIT. Print P # Go To Next	Get to start of data Next: Get next LAB. LAB found? If not EXIT. Get next project for this LAB with P # = P2 Project found? If not go to next. Print LAB # Go to Next 1.

Table 2.2 Two sample queries on the hierarchical model.

2-2.4 It can be observed, that even though the two original queries are symmetric in the sense that one is inverse of the other, the DSL procedures required to answer them are certainly not symmetric. This illustrates one of the drawbacks of the hierarchical model - i.e., 'unnecessary complexity. The hierarchical model also suffers from a number of anomalies in connection with storage operations of adding, deleting and updating, as indicated:

(a) Adding: It is not possible, without introducing a dummy laboratory, to insert data concerning a new project - say P7 - until it has been assigned to a particular laboratory.

(b) Deleting: If we deleted a laboratory having a particular project, the data concerning that project is lost, because deletion of any segment occurrence automatically deletes subordinate segments in keeping with the hiearchical philosophy.

(c) Updating: If we need to change the scope of a multi lab project, we are faced with either the problem of searching the entire data model to find every occurrence of the project or the possibility of rendering the data model inconsistant. The logical data base facility of IMS overcomes many of the difficulties, which in a way is a feature of the particular implementation and not the hierarchical approach (5).

2-2.5 Advantages

(a) It is a simple data model which provides the users with relatively few easy to master commands.

(b) Because of the constraints on type of relationships allowed, it can allow an easier implementation than other complex structures.

2-2.6 Disadvantages

(a) The restrictions imposed in using one-to many relationship, sometime causes unnatural organization. For instance, N:M relationship can sometime only be represented in an clumsy way.

(b) Because of the strict hierarchical ordering, operations such as insertions and deletion become quite complex.

(c) A delete operation can lead to loss of information present in the descendents, if null records are not permitted. Consequently, users have to be careful when performing a delete operation.

(d) It is sometimes not possible to answer symmetric queries easily in hierarchical system. Therefore, the structure of the data base may tend to reflect the needs of the application (7).

2.3 The Network Approach

2-3.1 The data base management system specifications as published in 1971 Report of the CODASYL Data Base Task Group (DBTG)[8], is a land mark in the development of data base technology. These specifications have been the subject of much debate and have served as the basis for several commercially available systems like DBMS/10, DMS-1100, IDMS and IDS/II.

2-3.2 Figure 2.2 shows how laboratories-and-scientific manpower might be represented in this system. The nodes of a DBTG network are

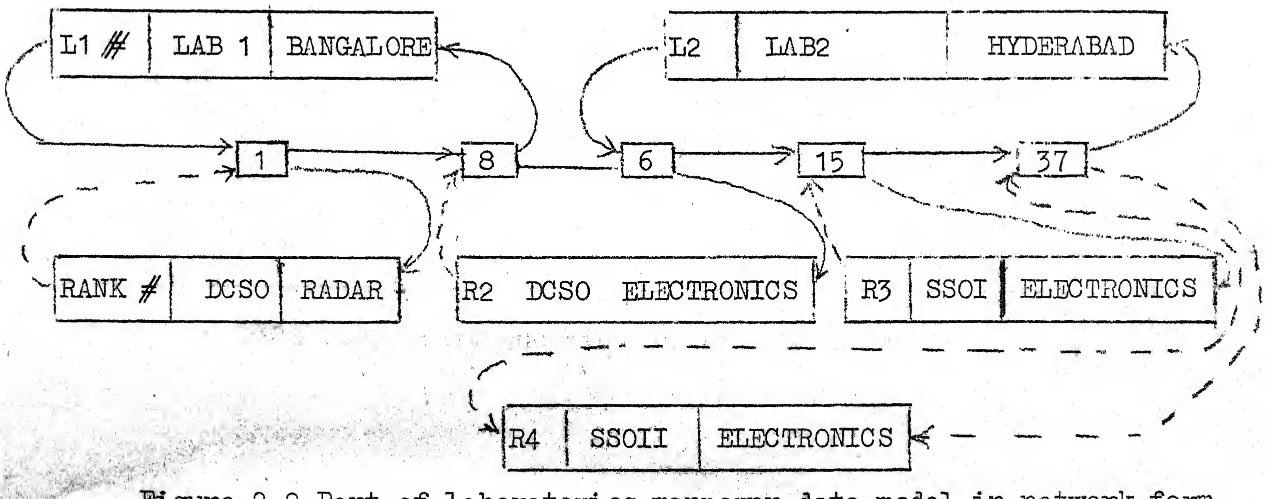


Figure 2.2 Part of laboratories-manpower data model in network form.

individual record occurrences. A network is a more general structure than a hierarchy because a given node may have any number of immediate superiors (as well as any number of immediate subordinates) - we are not limited to a maximum of one, as we are in a hierarchy. This enables us to represent a many-to-many correspondence in a reasonably direct manner. As shown in Figure 2.2, in addition to laboratories-manpower themselves, we introduce a third type of record which we may call LINK. A link record occurrence represents the connection between one laboratories and one grade/rank of scientific manpower, and contains data describing the connection (in this case the strength). All link occurrences for a given laboratory are placed on a chain starting at and returning to the laboratory. Similarly for all link occurrences for a given rank. Figure 2.2 shows chains for LAB1 and LAB2 and also incomplete chains for Ranks 1,2,3 and 4. The figure shows that LAB2 at Hyderabad has 6 PSCS, 15 SSOI and 37 SSO II.

Q1. Find the number of PScO in LAB2?	Q2. Find Lab # which has got 6 PScO?
Find laboratory with LAB # LAB2 Next: Find next link for the lab. Link found? If not EXIT. Find rank for this link. GET PScO Print Number Go to Next	Find Rank # for Name = PScO Next: Find next link for PScO Link found? If not EXIT. FIND Lab # for the link Get LAB PRINT LAB # GO TO Next.

Table 2.3 Sample querries on network model.

2-3.3 The data ~~language~~ sub language (DSL) for network model must obviously permit the user to traverse the various connecting chains. This is handled with the help of the 'FIND' statement. A 'Get' statement will retrieve the record occurrence just found. Considering similiar queries Q1 and Q2, the algorithm for answering the querries in network approach is given in Table 2-3. It can be observed that with the network approach, symmetric questions require symmetric answers - an advantage over the basic hierarchical approach. Storage anomolies are also overcome to a large extent as indicated below:

(a) Adding: It is trivial to add a new rank, say, JSO. Initially there will be no links for the new post; its chain will consist of single pointer from the part in itself.

(b) Deleting: We can delete any laboratory without llosing information about it and of rank structure of scientists.

(c) Updating: A feature of designation or number of persons in that cadre can be easily updated without inconsistency because it appears only at one place.

2-3.4 The major disadvantage of the network model is simply that it is too close to storage structure. The user has to be thoroughly aware of which chains do and do not exist, and his DSL programming rapidly becomes extremely complex. More significantly chains are directly visible to the user and hence must be directly represented in storage. There is thus a risk in the network approach that the user will become locked into a particular storage structure, contrary to the aim of data independence.

2-3.5 A partial implementation of CODASYL DBTG report has been carried out as an extension to FORTRAN at Purdue University on CDC-6500(6). Most of DDL, search routines and DML features have been achieved on the system using subroutine calls.

2.4 <u>The Relational Approach</u>

2-4.1 As James Martin writes in (1), "Through out the history of engineering a principle seems to emerge: "<u>Great engineering is simple engineering</u>." Ideas which become too cumbersome, inflexible, and problematic tend to be replaced with newer, conceptually cleaner ideas which compared to the old, areaesthetic in their simplicity. E.F. Codd in his paper of 1970(9) introduced a simple and ingenious concept which set the direction for research in relational data model for DBMS. The paper defined data sublanguage as a set of facilities, suitable for embedding in a host programming language, which permits the retrieval of various subsets of data from a data bank. The paper noted that a standard logical notation, the first order predicate calculus, is appropriate as a data sublanguage for n-ary relations. The paper also 'introduced a set of operator like 'Join', 'Projection' etc., which were later developed into the well known relational algebra. Finally the paper explored the properties of 'redundancy' and 'consistency' of relations, which laid the ground work for Codd's later theory of normalisation(10).

2-4.2 <u>Definitions</u>: In mathematics, the term Relation is defined as follows:

"Given sets D1, D2, ..., Dn (not necessarily distinct), R is a relation on these n sets, if it is a set of ordered n-tuples $d_1, d_2, \ldots, d_n$; such that d_1 belongs to D_1, d_2 belongs to D_2 ..., d_n belongs to D_n.

Sets D_1, D_2, ..., D_n are called domain of R. The number n is called the degree of R and the number of tuples is cardinality." (5)

2-4.3 Table 2.4 indicates one such relation PROJ. It is customery (though not essential) when discussing relation to represent a relation as a table in which each row represents a tuple. In the tabular representation of a relation the following properties, which derive from the definition of relation, should be observed:

(a) No two rows (tuples) are identical.

(b) The ordering of rows is not significant.

(c) The ordering of column is significant. However, if any individual column is referred by the appropriate domain name, never by its relative position, then the ordering of column is insignificant.

(d) Every value within a relation is an atomic data item (Normalised discussed in Chapter III).

2-4.4 The columns of the table are called attribute. If two or more columns have the same domain name, a seperate role name is prefixed to domain name depending on its role. The individual entries like 'LRDE'

CODE	QR NO	TYPE	NAME	LAB	PR NO
1	1	1	FA RADAR	LRDE	LRD-32
2	5	2	DOP Radar	LRDE	LRD-76
⋮	⋮	⋮	⋮	⋮	⋮
1001	15	1	GPA FOR WSC42	DLRL	DLR-18
1002	25	2	DF AREIL	DLRL	DLR-29
⋮	⋮	⋮	⋮	⋮	⋮

Table 2.4 Tabular Relation PROJ.

are called its components. A column or set of columns whose values uniquely identify a row of a relation is called a candidate key or simply key. When a relation has more than one key it is customary to designate one as the Primary Key. In relation PROJ, [illegible], CODE is the Primary key.

2-4.5 Languages: There is large variety of languages for relational DBMS - four classes of which i.e., relational calculus, relational algebra, mapping oriented language and graphic-oriented language are discussed briefly.

(a) Relational calculus: The relational family of languages grew from the observation that the first order applied predicate calculus can be used as DSL for normalized relation. CODD presented the details of such a calculus based sublanguage called DSL ALPHA(11). A typical query in ALPHA has two parts; a target, which specifies the particular attributes, which are to be returned and a qualification, which selects particular tuples from the target relation by giving a condition which they must satisfy.

Example: Get W (PRON. NAME, PROJ. LAB): PROJ. CODE = '1001'

Result W

NAME	LAB
GPA FORWS C42	DLRL

Mathematically, DSL ALPHA has been proved to be a complete language. Various other languages based on relational calculus are QUEL(12), COLARD(13) and RIL(14). The relational calculus languages provide the undermentioned distinct advantages over IMS/DDLI and DBTG DSL.

(i) Provides complete data independence, as it contains no reference to storage/access details.

(ii) the language is very simple yet complete.

(iii) The language is non procedural.

(iv) The retrieval power of basic language can be simply and indefinitely extended by providing library functions.

(v) It supports higher level languages.

(b) Relational Algebra: The relational algebra is a collection of operators that deal with whole relations, yielding new relations as a result. The major operators of relational algebra include:

(i) Projection operator: returns only the specified column of given relation and eliminates duplicates from the result.

(ii) Restriction operator selects only those tuples of a relation which satisfy a given condition.

(iii) Join operator takes two relations as arguments and a new relation is formed by concatenating a tuple from each, wherever a given condition holds between them.

(iv) Set-theoretic operators like union, intersection, and set difference take two relations as operands, treating each as a set of tuples, and produce a single relation as a result - provided the operand relations have compatible sets of attributes.

(v) Division operator which operates on two input relations to produce a third relation.

(vi) Nesting of operators. The algebra has the convenient property that its operators can be nested to form expression of arbitrary complexity by using parenthesis.

Languages based on relational algebra have been implemented at MIT(15), IBM Scientific Centre, England(16), and General Motors Research Laboratory(17). As compared to DSL ALPHA, the languages based on relational algebra provide data independence, simplicity, nonprocedurality, ease of extension and support for higher level language, to a lesser degree.

(c) Mapping-Oriented Language: These languages directed at the non-programming professional, offer power equivalent to that of the relational calculus or algebra, while avoiding mathematical concepts such as quantifiers. Mapping oriented languages include SQUARE, a terse APL notation;(18) SEQUEL, a structured language based on English keywords(19), and SLICK, a language intended for implementation on associative hardware(20). The basic building block of such languages is the 'mapping' which maps a known attribute or set of attributes into a desired attribute or set of attributes by means of some relation.

(d) Graphic-Oriented Languages: In recently developed languages, the user states his query not by conventional linear syntax, but by filling in blanks or making a choice on a graphic display. Examples of such language are CUPID(21), and Query by Example(22).

2-4.6 Storage Anomalies: As far as storage operations are concerned, it is sufficient to observe that in relational approach no difficulties/anomalies arise provided correct normalised relations have been chosen. Considering the same operation:

(a) Adding: It is trivial to add a new project say 1005. Doing so will involve adding a new tuple to PROJ relation;

(b) Deleting: We can delete any tuple or even a complete relation without losing any information.

(c) Updating: We change PRNO in any tuple without search problems and without the problem of inconsistency because PRNO appears precisely in one tuple.

2.5 Comparative Evaluation of the Approaches

2-5.1 The comparison of the three approaches for Data Models from a number of points has been done - there is no simple answer to the best choice. Each approach, will meet the needs of a portion of diverse users community. CODASYL DBTG implementations are being marketed commercially. Relational implementations are being developed in university and research laboratories. There is no fully implemented Relational DBMS at present.

2-5.2 In hierarchical approach, relationships are entirely implicit. The relationship between two entities is represented in someway by a relative position occurrence of the concerned sequence. As a consequence the corresponding data sublanguage relies heavily on the concept of position within the data base : in fact the DSL must provide two types of operations, one for positioning and one for retrieving or storing, although the distinction may not be very clear in a real system. The only way a user can retrieve the information contained in a relationship is by building the information up dynamically as the result of sequence of DSL operations.

2-5.3 In network approach on the other hand, relationships are represented explicitly by means of pointers. However, the very fact that pointers are used means that the relationship and entities are considered as difficult things. Again the DSL must provide two types of operations, and again retrieving information from a relationship involves building up of information dynamically.

2-5.4 In relational approach, relationships are again represented explicitely. Here, however, they are represented in exactly the same way as the entities i.e., by means of tuples. In other words, relationships and entities are considered as the same type of object in relational approach. It is thus possible to provide a DSL with a uniform set of operations for manipulating both - an obvious simplification. It can be argued, in fact, that the relational model is a views of data base in terms of its natural structure only : it contains absolutely no consideration of storage access details such as position or pointers, it contains no 'representation clutter.'

2-5.5 In view of the above, it was decided to base DBMS implementation on the Relational approach, using the feature of DSL based on Relational calculus - such as DSL ALPHA.

CHAPTER III

RELATIONAL APPROACH - DESIGN DETAILS

3.1 General

While discussing properties of a relation in Section 2-4.4, an important property was enumerated without details. It is the property that 'every value (component) within a relation is an atomic data item' This property has come to be known as Normalization. In this chapter, it is proposed to discuss Normalisation concepts, broad system design and the data structures used in the implementation of the Relational Approach.

3.2 Normalization

Normalization theory begins with the observation that certain collections of relations have better properties in an updating environments than do the other collections of relations containing the same data. The theory then provides a rigorous discipline for the design of relations which have favourable update properties. The theory is based on a series of Normal Forms - first, second and third form - which provide successive improvements in the update properties of the data base (see Figure 3.1).

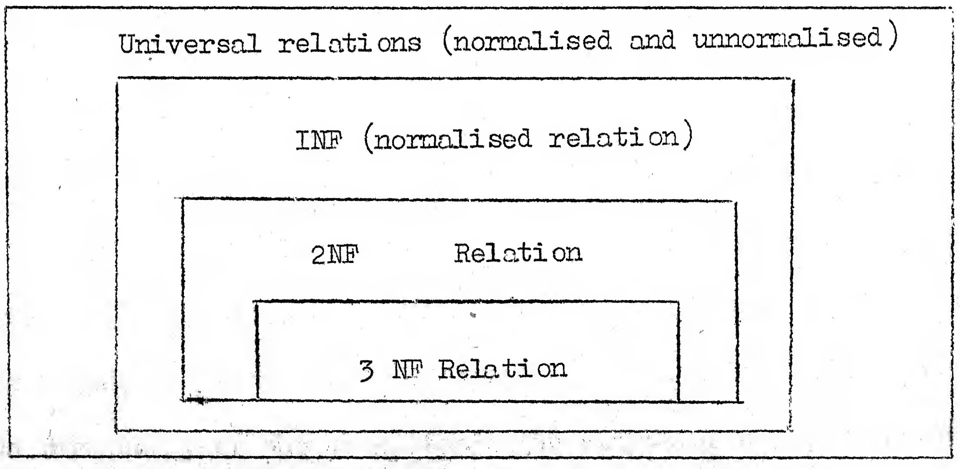

Universal relations (normalised and unnormalised)
1NF (normalised relation)
2NF Relation
3 NF Relation

Figure 3.1 Levels of Normalisation.

(a) First Normal Form (1NF): A relation in first normal form is a relation in which each component of each tuple is non-decomposable, i.e., the component is not a list or a relation. The relations in first normal form are sometime called "flat tables". As discussed in Chapter II, a relation in first normal form may exhibit the three kinds of anomolies or misbehaviours called insertion, deletion and update anomalies.

(b) Second Normal Form (2NF): A normalized relation R is said to be in 2NF (second normal form), if and only if the nonekey domains of R, if any, are functionally dependant on the primary key of R. Relation in 2NF exhibit improved performance as regards the anomalies. 2NF is of little significance except as a stopping off place on way to 3NF.

(c) Third Normal Form (3NF): A relation R is said to be in 3NF if it is in 1NF and, for every attribute collection C of R, if any attribute not in C is functionally dependent on C, then all attributes in R are functionally dependent on C(23). Sharman gives a simple definition as "a relation is in 3NF if every determinant is a key"(24).

Both definitions are formal ways of expressing a very simple idea: that each relation should describe a single 'concept' and if more than one concept is found in a relation, the relation should be split into smaller relations. The design of a data base in 3NF depends on a knowledge of the functional dependencies among the attributes of the data. The knowledge cannot be discovered automatically, but must be furnished by the Data Base designer who understands the semantics of the information.

In fact there is no unique 3NF representation of a given data base. Keeping in view the ease with which update, insertion and deletion anomalies are avoided by 3NF, it was decided to implement relational data base using relations in 3NF. The relations will have to be converted into 3NF by the Data Base Designer manually at the Data Definition stage. Functional dependencies in relation PROJ, which is in 3NF is shown in Figure 3.2. It may be observed that non-key domains are

(a) mutually independent;
(b) functionally dependent on the primary key CODE.

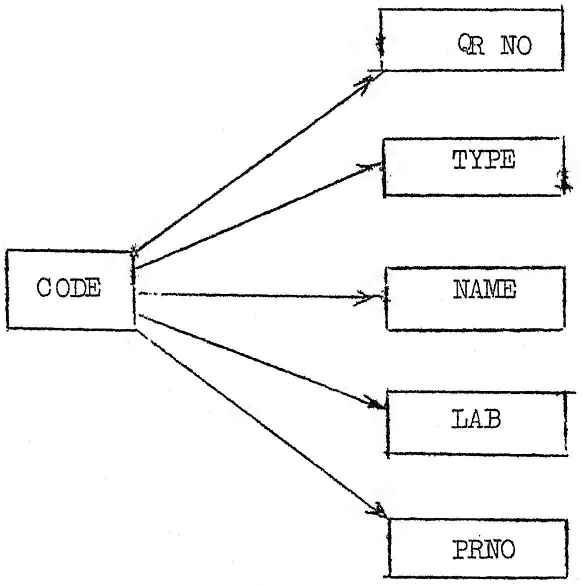


Figure 3.2 Relation PROJ function dependencies.

3.3 Overall System Design

Figure 3.3 shows the Block Schematic of the overall system view of the Relational Data Base Management System being implemented. Brief description of each of the blocks is given below:

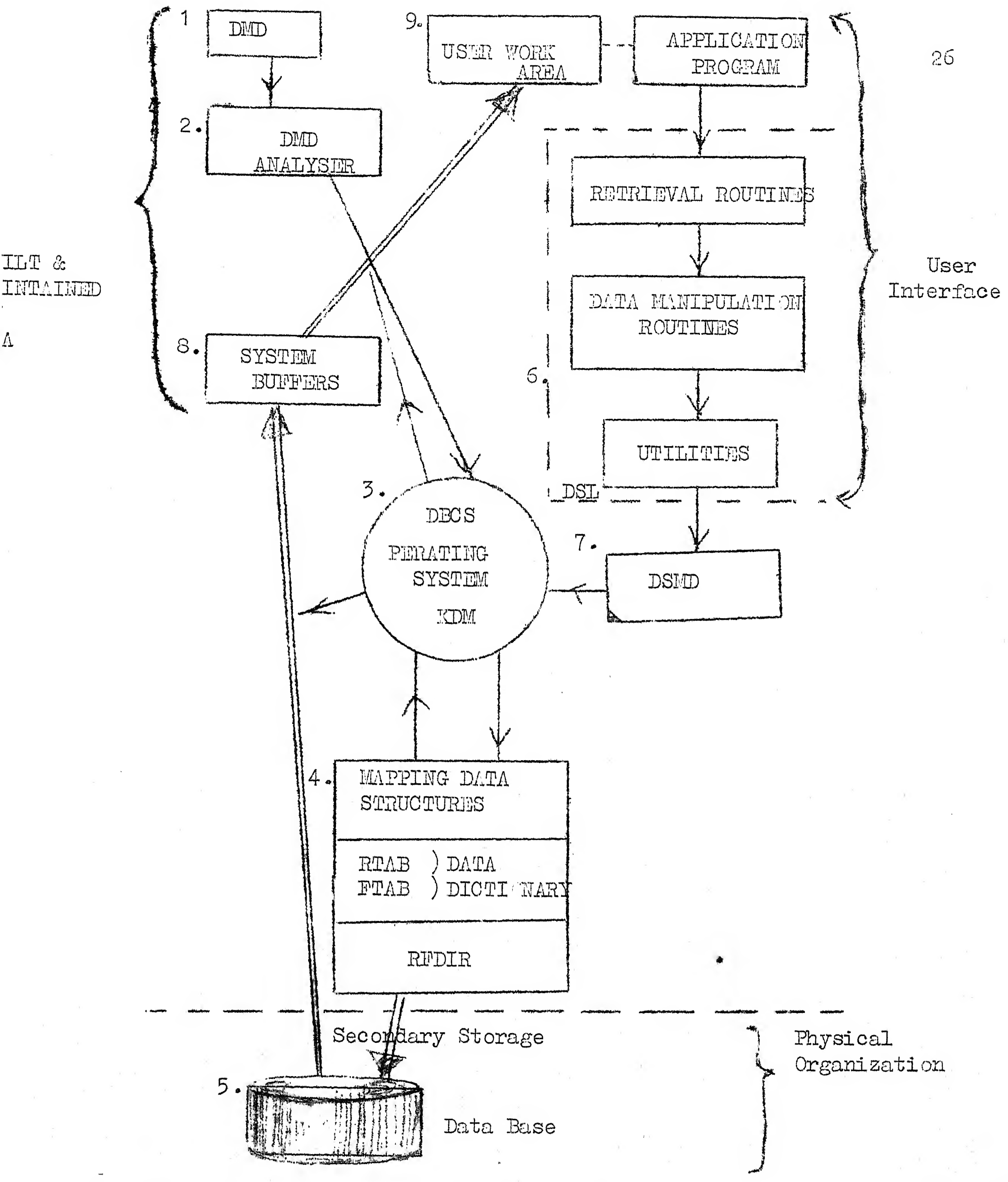


Figure 3.3: Overall Block Schematic.

3-3.1 Data Model Definition (DMD): DMD includes definition of all the relations and their respective domains. For simplicity, first domain in each of the relation will normally be the KEY. The following commands have been used to form Data Model Definition Language, which can be given on KBD or CR on TDC-316:

(a) RELNƀƀNAMEƀƀAC01	- The command defines a relation, its NAME and Access code and Level.
(b) FLDSƀƀNAMEƀƀIN F ORMATS FIXED IN-INTEGER I5 AN-ALPHA10A2 AD-ALPHA30A2	- The command defines a field or item name, and its type INTEGER, ALPHANUMERIC (SINGLE or DOUBLE). For simplicity fixed formats have been used.
(c) DATAƀƀ99	- This command indicates that all ITEM names have been completed. Data tuples follow. The number of tuples to be included in the relation are given for definition purposes.
(iv) IEND	- The command indicates the end of DMD and completion of Data Definition phase by the DBA.

3-3.2 DMD Analyser: This is the main software which checks the validity of each of the commands of DMD - builds up various tables as part of the data structure module. It also prepares the INDEX for the various storage devices and places the relation on physical devices in conjuction with the Data Base Control System. The implementation of DMD Analyser has been discussed in details in Chapter IV.

3-3.3 Data Base Control System: TDC-316 does not support multi-programming as such the software has been developed for single sequential working. The module encompasses the system operating system which includes key to disk management (KDM) system, FORTRAN compiler and run time routines including library functions. The KDM software provided by ECIL-Hyderabad does not support REWIND command in FORTRAN for the disk. It was decided to modify the system to provide REWIND command - as it would lead to saving in memory space especially for data manipulation routines. The existing system automatically rewinds as and when READ/WRITE commands are changed. It is desired to eliminate this feature - when REWIND command works. Accordingly, the system was modified and the patches to the run time control routine for FORTRAN (FC) and KDM are given at Appendix 'A'.

3-3.4 <u>Data Structures for Mapping</u>: For mapping the logical structure of data base system on to the physical organization there is a requirement of various data structures in the form of system table - which are required to be built and maintained by DBMS. Following are the important structures.

(a) <u>Relation Table (RTAB)</u>: The table includes one tuple for each of the relations in the system and stores the detailed information about the relation. The layout of RTAB is given in Figure 3.4. The table is stored as an array (20,10) in the system to cater for 20 relations in the system.

RTAB

Relation ID	Name of the Relation	Device No	Starting Position	Access Control	No. of tuples	No. of formats		
1	2,3	4	5	6	7	NI 8	NA 9	AD 10
1	PROJ	4	1	AB	10	3	3	0
	NR	Pointer to current relation.						

Figure 3.4 RTAB Parameters

(b) Item or Field Table (FTAB): The item table forms a part of Data Dictionary, which maintains a list of all the item names, their types and identification used. Layout of the FTAB is given in Figure 3.5. The table is stored as an array (400,4) in the system to cater for 400 distict items or field names.

FTAB

Item or Field ID	Name	FORMAT
1	CODE	1 = IN FORMATS 2 = AN CODE 1 3 = AD used.
NF	Pointer to current item/field	

Figure 3.5 FTAB Parameters with a Sample entry.

(c) Relation Field Directory (RFDIR): This is a data structure which contains the list of all the relation ID's and their corresponding field or item ID's. For ease of manipulation FORMAT as for each of the item is also stored in the same items. RFDIR layout is given in Figure 3.6. The table is stored as an array (50,3) to cater for 50 entries.

RFDIR

Relation ID	Field or Item ID	FORMAT
1	1	1
NFR	Pointer to current entry	

Figure 3.6 RFDIR Parameters with a Sample entry.

(d) Matrix of Common Field In Relations (MCFR): Based on the common items or fields between different relations a nxn matrix is formed, if user wants to use n relations. This matrix MCFR is used by various search and retrieve routines for transferring control from one relation to the other. MCFR has been represented as (10,10) array to cater to 10 relation usage at a time.

3-3.5 Secondary Storage: This is the physical organization on which the main data base is stored. The existing disk unit with TDC-316 has the following features:

(a)	number surfaces	10
(b)	number tracks/surface	200
(c)	number sectors/track	10
(d)	capacity per sector/track	128 words

Thus the system having moveable head provides a capacity of 7.8M byte. With the help of KDM version 1, the disk has been divided into the following logical units - and accepts macro commands like READ, WRITE and REWIND.

Track No.	Device Designation	Device No.
1-31	Not assigned for System Use	
32-49	D4	13
50-99	D1	14
100-149	D2	15
150-199	D3	16

In DMD and other software device numbers (DNO) 2,3,4 and 5 have been used in the index for 13, 14, 15 and 16 respectively.

3-3.6 Data Sub Language Routines (DSL): This is the major software developed which provides data sub model definition as user view for each application. The various routines can be called from any application program, along with any library functions. Each of these routines have been discussed in details in Chapter IV.

3-3.7 Data Sub Model Definition (DSMD): With the help of DSL a limited logical sub model indicating users logical view is built up during definition phase.

3-3.8 System Buffers: BUFF (2,40) an array has been assigned as system buffer to hold all interim values for manipulation by different routines.

3-3.9 User Work Area: IUSR (40,40) an array has been assigned as user's work area for holding all the output for the user till he wants them on a particular I/O device.

3.4 Operating System

The relational data base can be implemented in the following two phases; which are independent.

(a) Phase 1: Data Model Definition phase for building and maintenance of data base by the DBA. Once all the relations have been generated the contents of the data structures are automatically placed in the system area on disk for use in Phase 2. The programs can be erased, after use.

(b) Phase 2 (DSL Phase): First the software tables are fetched from system area on disk whenever user OPENS the data base. His access code and level is checked and his authorisation for the relations requested is verfied before building up MCFR. With the help of various retrieve and data manipulation routines the desired output is obtained. After the use is over data base is CLOSED to initialise the various structure in the system. In case of errors/failures DUMP can be requested by users with level upto 10.

CHAPTER IV

IMPLEMENETATION OF DBMS ON TDC-316

4.1 General

Keeping in view the limited memory available with FORTRAN run time routine it was decided to attempt a partial implementation of Relational DBMS, covering the essential features. Once the principle is proved it would be possible to modify certain rigid features and dimensions to accomodate extra features. In this chapter DMD analyser and DSL subroutines are discussed in detail.

4.2 DMD Analyser Modules

Data Model Definition Analyser module is required to analyse DMD Language commands and accordingly build the relations and associated data structures. The main functions of the module are --

(a) Check proper authorisation of DBA for building the data base.

(b) Verify DMD language commands shown in Section 3-3.1.

(c) Based on the commands, CALL the appropriate routines for house keeping.

(d) Build up system data structures RTAB, FTAB and RFDIR.

(e) Allocate disk space to the relations and read in all the data.

(f) Cater for 10% extra space for addition of tuples.

(g) Once all the relations data has been read in, store the system data structures in system area for use by DSL.

The DMD main and its associated subroutines linkages are shown in Fig. 4.1. Let us discuss the functions, working and flowchart of each of the constituents.

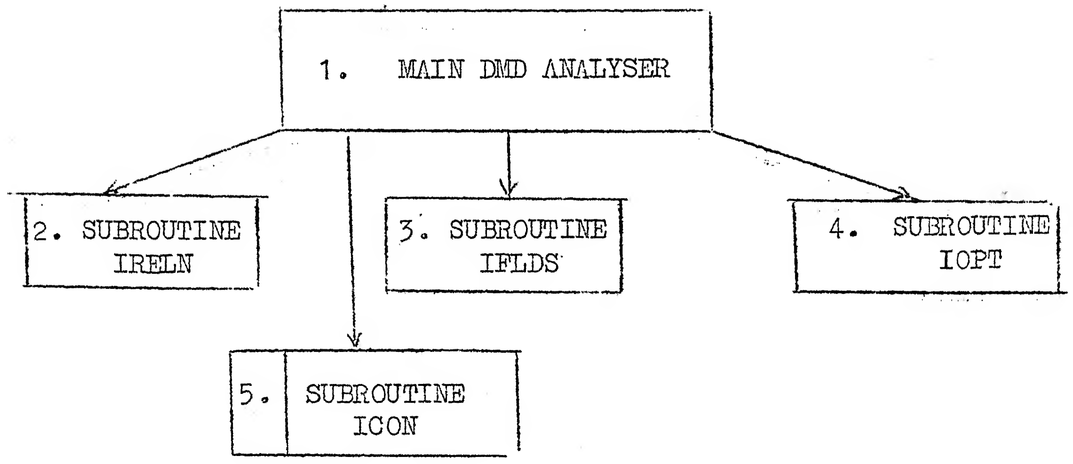


Figure 4.1 DMD Analyser module with it s subroutines.

4 4-2.1 Main DMD Analyser: The flow chart for Main DMD Analyser is shown in Figure 4.2. Functions performed by the program are -

(a) Initialisation of all tables, pointers and variable codes.

(b) Validating RELN command, storing relation name and its access code.

(c) Validating FLDS commands and storing field name and type in FTAB for all items.

(d) Testing for DATA command and ascertaining number of tuples in the initial relation. Working out the extra space for adding tuples during run time. ICON routine is used for converting from A2 to I5 format.

(e) Data structure RTAB is built up after disk storage details have been worked out.

(f) Subroutine IOPT is used for input/output from any device to system buffers and from system buffer to the appropriate device. All the relation tuples are placed on the secondary storage and an index is maintained in RTAB.

(g) Tests for IEND command after each relation - if not, it checks for a fresh RELN card and repeats (b) to (f) till IEND command is encountered.

(h) All system tables RTAB, FTAB and RFDIR are stored in system area (D4 or Device 16).

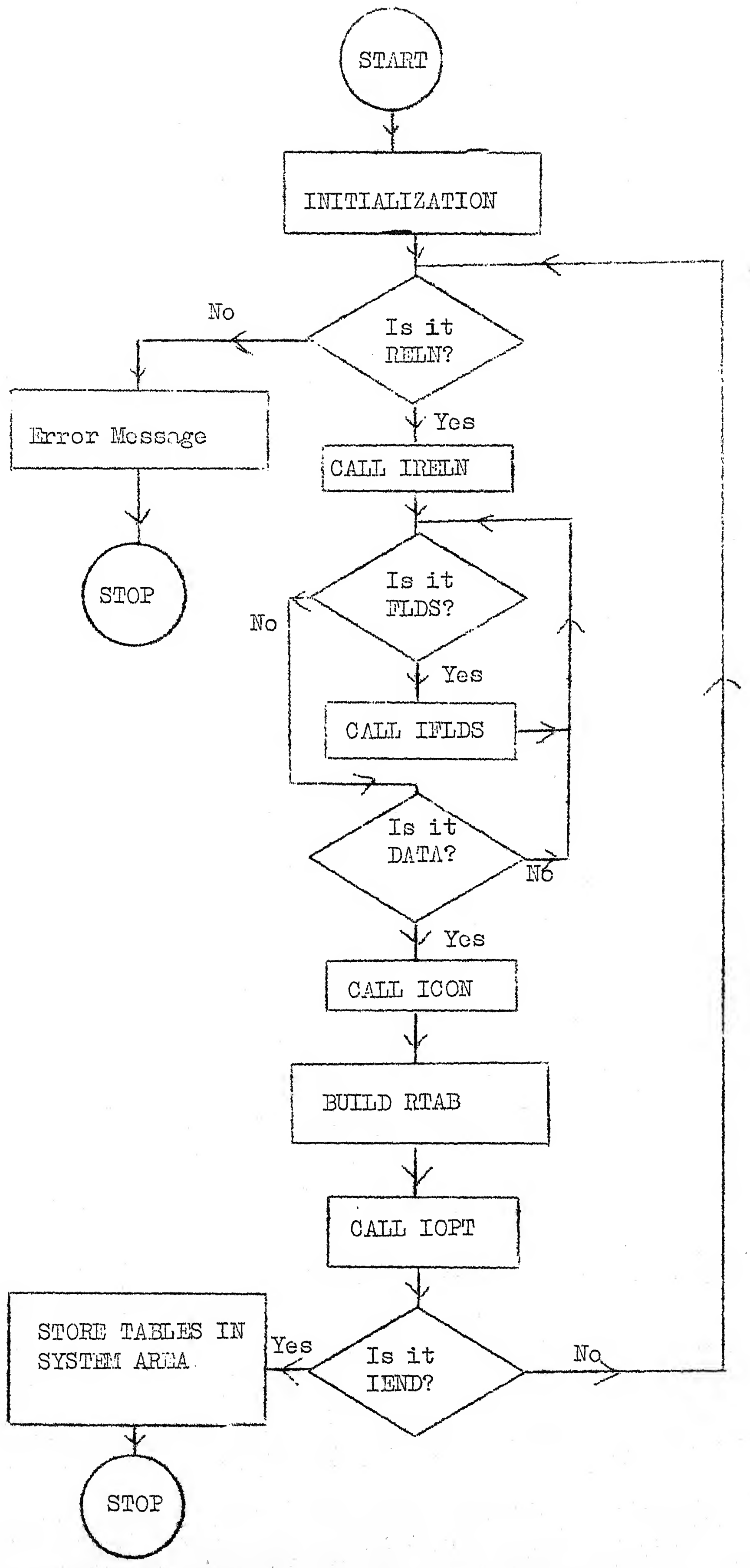
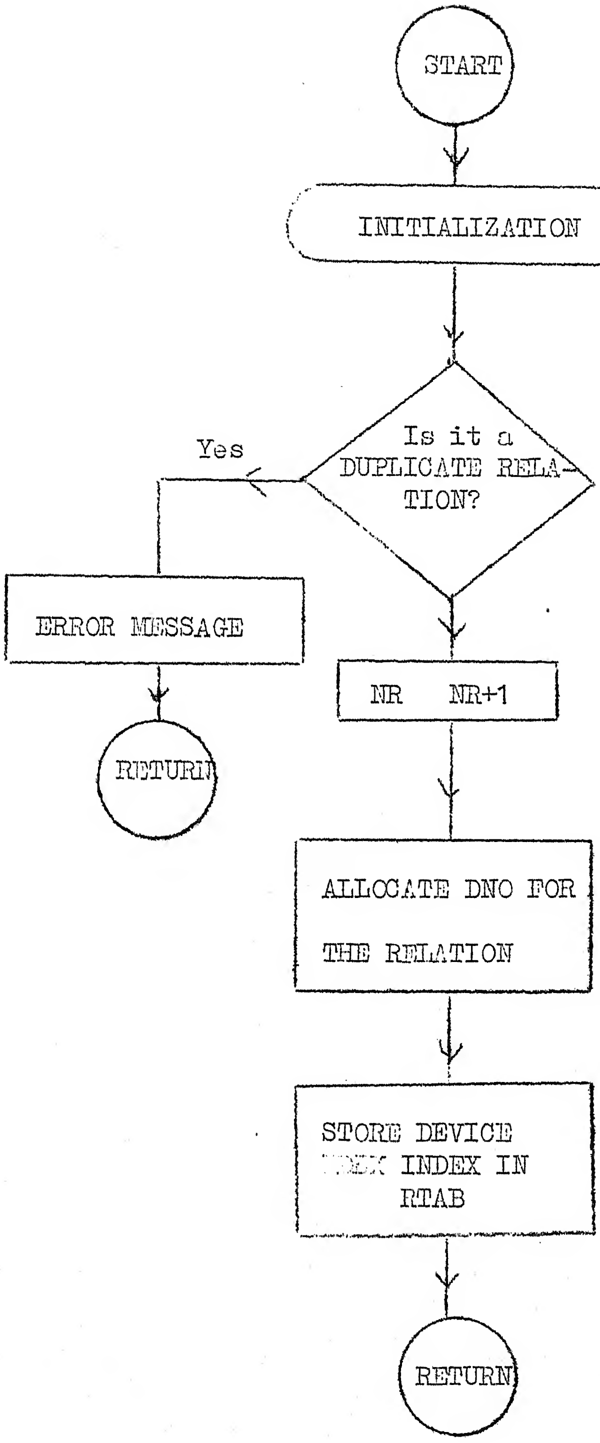


Figure 4.2:Main DMD Analyser Flowchart.



START
INITIALIZATION
Is it a DUPLICATE RELA- TION?
Yes
ERROR MESSAGE
RETURN
NR NR+1
ALLOCATE DNO FOR THE RELATION
STORE DEVICE INDEX INDEX IN RTAB
RETURN

Figure 4.3: Flowchart for Subroutine IRELN.

4-2.2 Subroutine IRELN: Flow chart for subroutine IRELN is given in Figure 4.3. The following functions are performed by IRELN:

(a) Initialisation of NI, NA and DA when the new relation is encountered.

(b) Testing relation name, if a relation with same name has earlier been defined-give an error message.

(c) Increment the pointers.

(d) Allocate a suitable device number, depending on how many relation have been defined so far.

(e) Maintain an index of the relations on a particular device and their relative location.

4-2.3 Subroutine ICON: Since DMD language commands are read in A2 format, subroutine ICON is used to convert integer values read in A2 format to I5 format for manipulation of integers. In TDC-316 in A2 format AB is stored as BA; each character occupying 8 bits. ICON, unpacks the characters and depending on their codes works out the corresponding integer value.

4-2.4 Subroutine IFLDS: The flow chart for subroutine IFLDS is given in Figure 4.4. The subroutine performs the following functions:

(a) Increments pointer both for FTAB and RFDIR.

(b) Testing, whether the item or field name has earlier been defined to enter the proper FLDID.

(c) If it is a new field name a fresh FLDID is allotted and record kept in FTAB, which helps as Data Dictionary.

(d) The type of formats used are given below:

Format	Alphanumeric Code	Integer Code
I5	IN	1
10A2	AN	2
30A2	AD	3

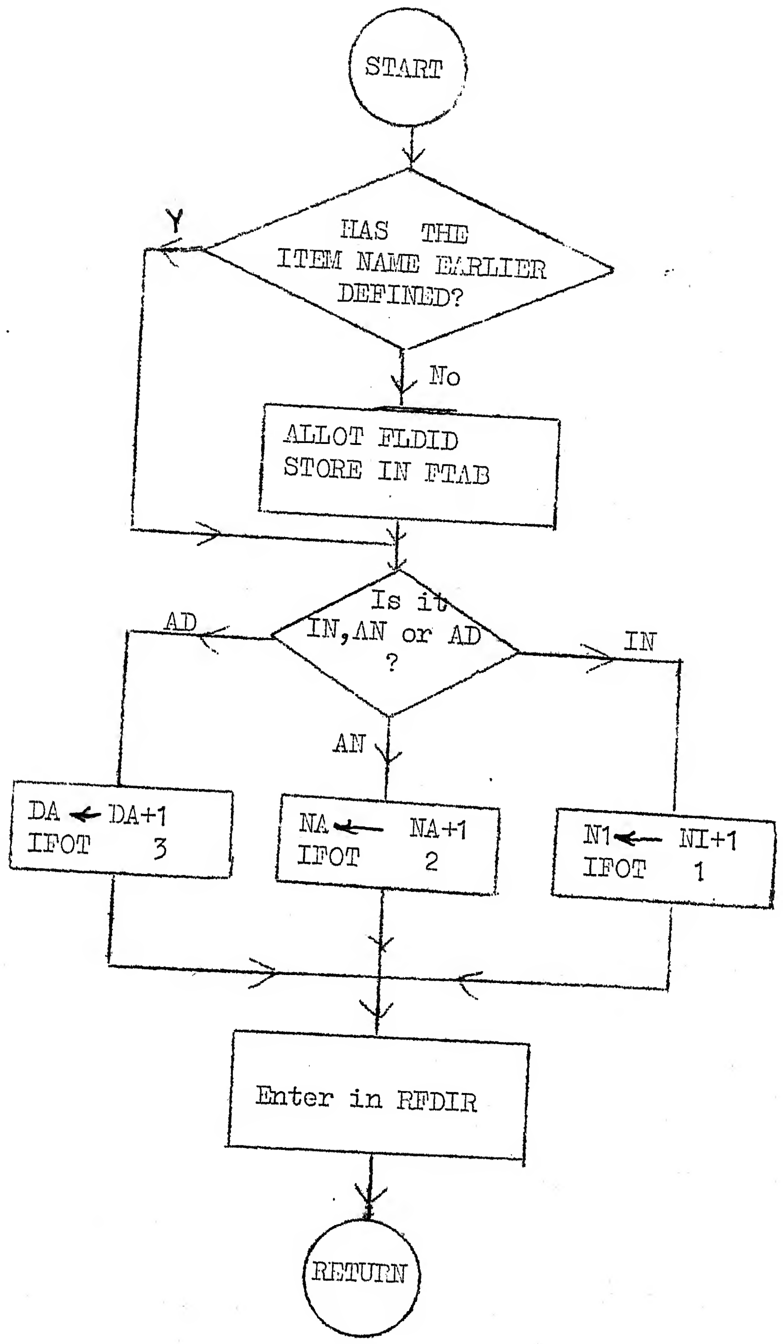


Figure 4.4: Subroutine IFLDS Flowchart.

(e) Appropriate type is recognised and corresponding entry made in RFDIR.

(f) Considering the type of applications it was felt that any information could be converted into the above three formats without any loss of information.

4-2.5 Subroutine IOPT: This is a general purpose subroutine to provide all types of input and output facilities on different devices. TDC - FORTRAN provides a very flexible N-Format, which has been used to provide any combination of IN, AN or AD formats[25]. BUFF(10,40) is the system buffer used by this subroutine. The same subroutine is also used in Data Sub Language routines for input/output facility. Thus the routine also tests, if your level corresponds to read only or both read and write access. If a read only user wants to write in the data base - an appropriate error is indicated and access terminated.

4.3 Data Sub Language Program Modules

As brought out in Chapter III, data sub language for the relational DBMS consists of a number of utility routines, search routines and data manipulation routines. Figure 4.5 indicates the various subroutines which can be called from users program after the data base ~~has~~ been opened properly. By and large the routines cover the various feature of DSL ALPHA, a sub language suggested by CODD - based on relational calculus. Let us discuss each of the subroutines in details.

4-3.1 Subroutine OPEN: This is the first routine which must be called by the user before he can work with the data base. Figure 4.6 gives the flow chart of the subroutine. Its main functions are:

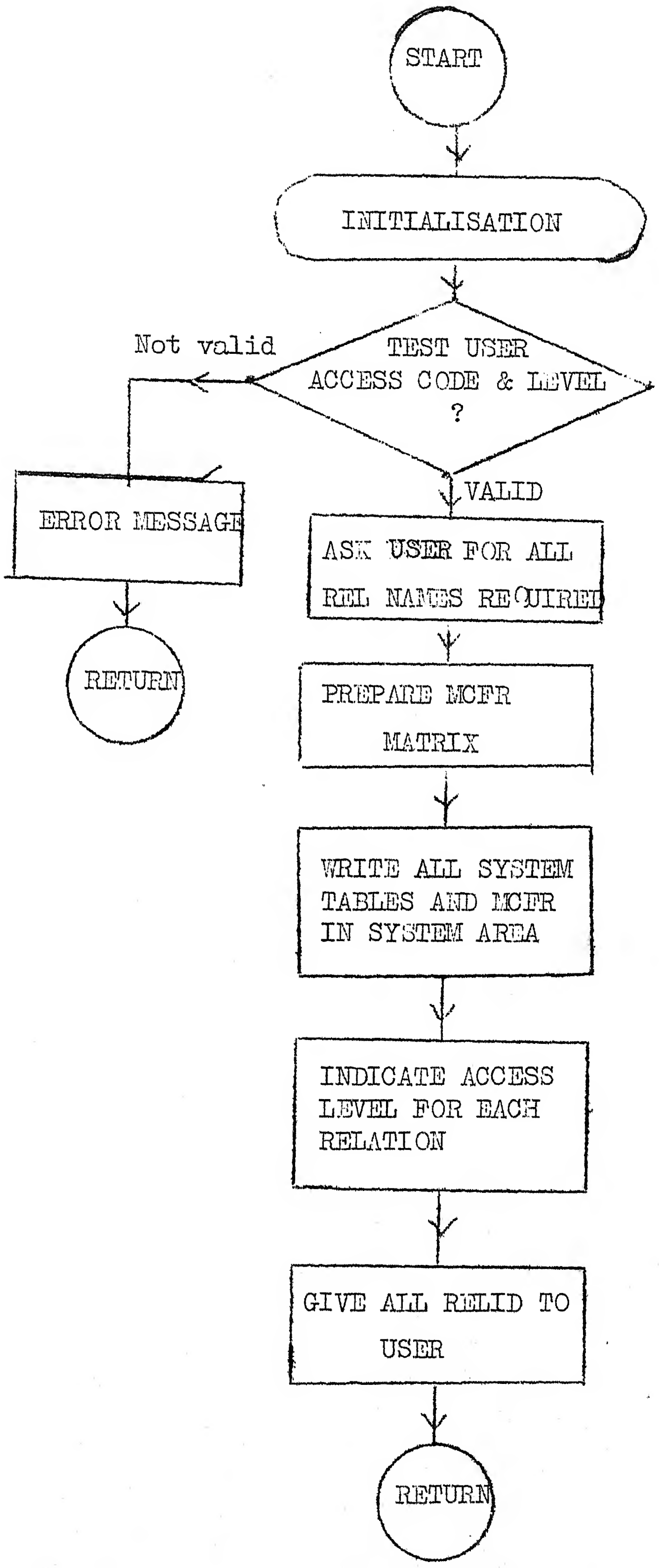

START
INITIALISATION
TEST USER ACCESS CODE & LEVEL ?
Not valid
ERROR MESSAGE
RETURN
VALID
ASK USER FOR ALL REL NAMES REQUIRED
PREPARE MCFR MATRIX
WRITE ALL SYSTEM TABLES AND MCFR IN SYSTEM AREA
INDICATE ACCESS LEVEL FOR EACH RELATION
GIVE ALL RELID TO USER
RETURN

Figure 4.6: Subroutine OPEN flowchart.

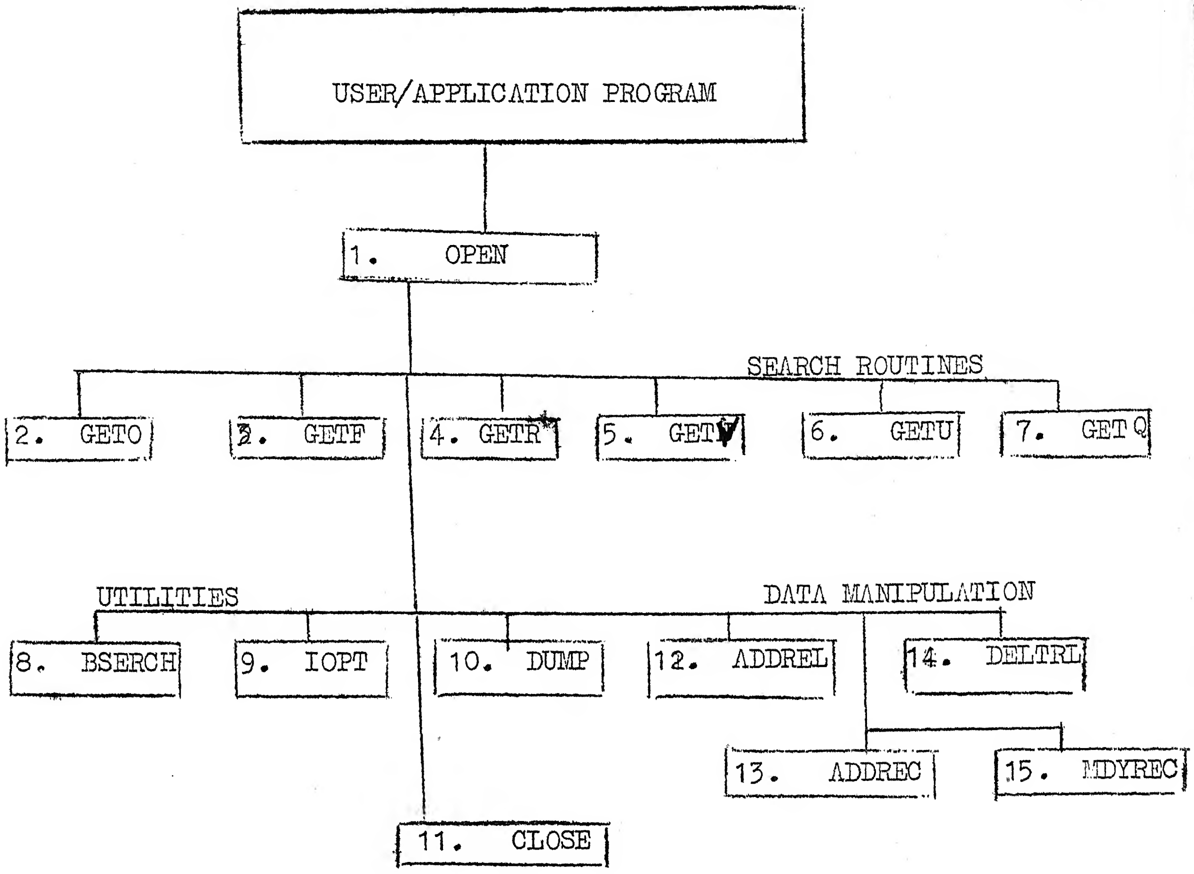


Figure 4.5 Data Sub Language Routines and their inter connection.

(a) Brings all the system tables from system area to the main memory, since this works independent of DMD Analyser.

(b) Asks user for this access code and level for validity.

(c) Checks level and flags when he is a Read only user.

(d) For other search and utilities routine, there is a requirement to find at least one common field between any two relations. If there is no common field - the particular relation cannot be used in continued queries involving two or more relations. To avoid time wastage in other routines a common field relation matrix MCFR is prepared after checking common items in all the relations requested by the user.

(e) The system table and MCFR are stored on disk for subsequent use.

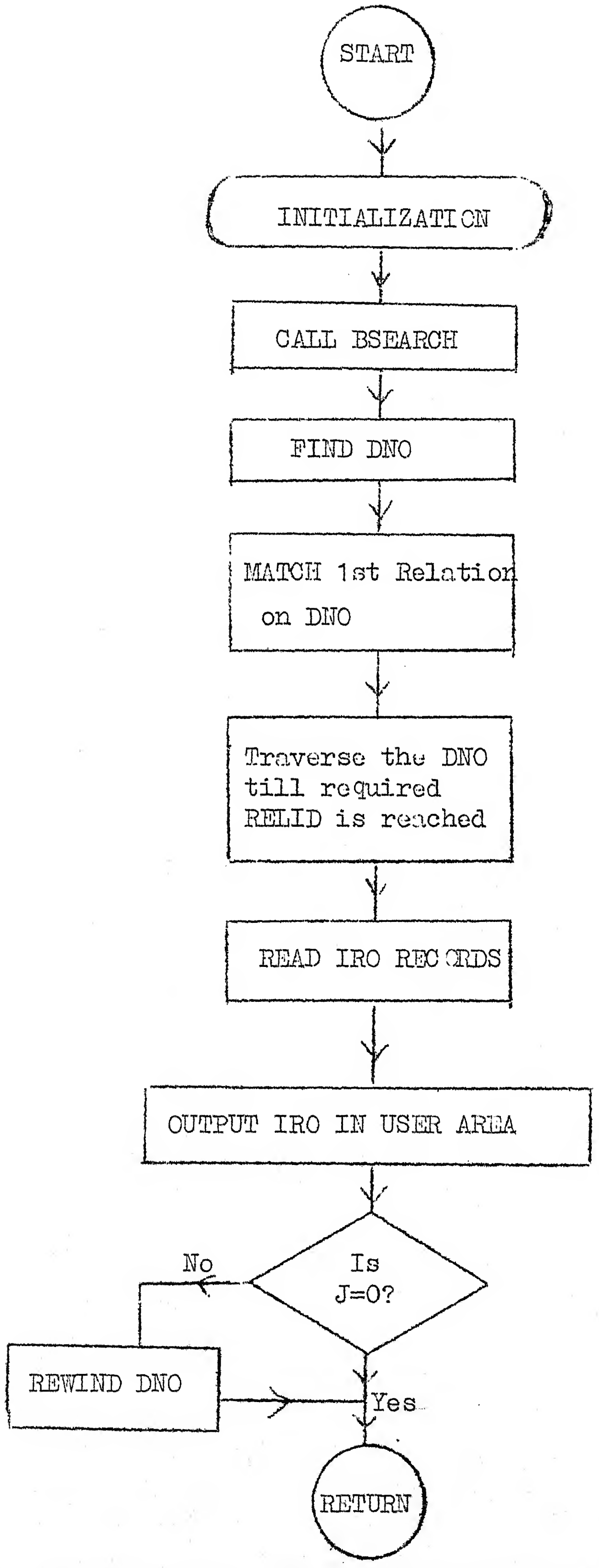


Figure 4.7: Flowchart Subroutine GETO.

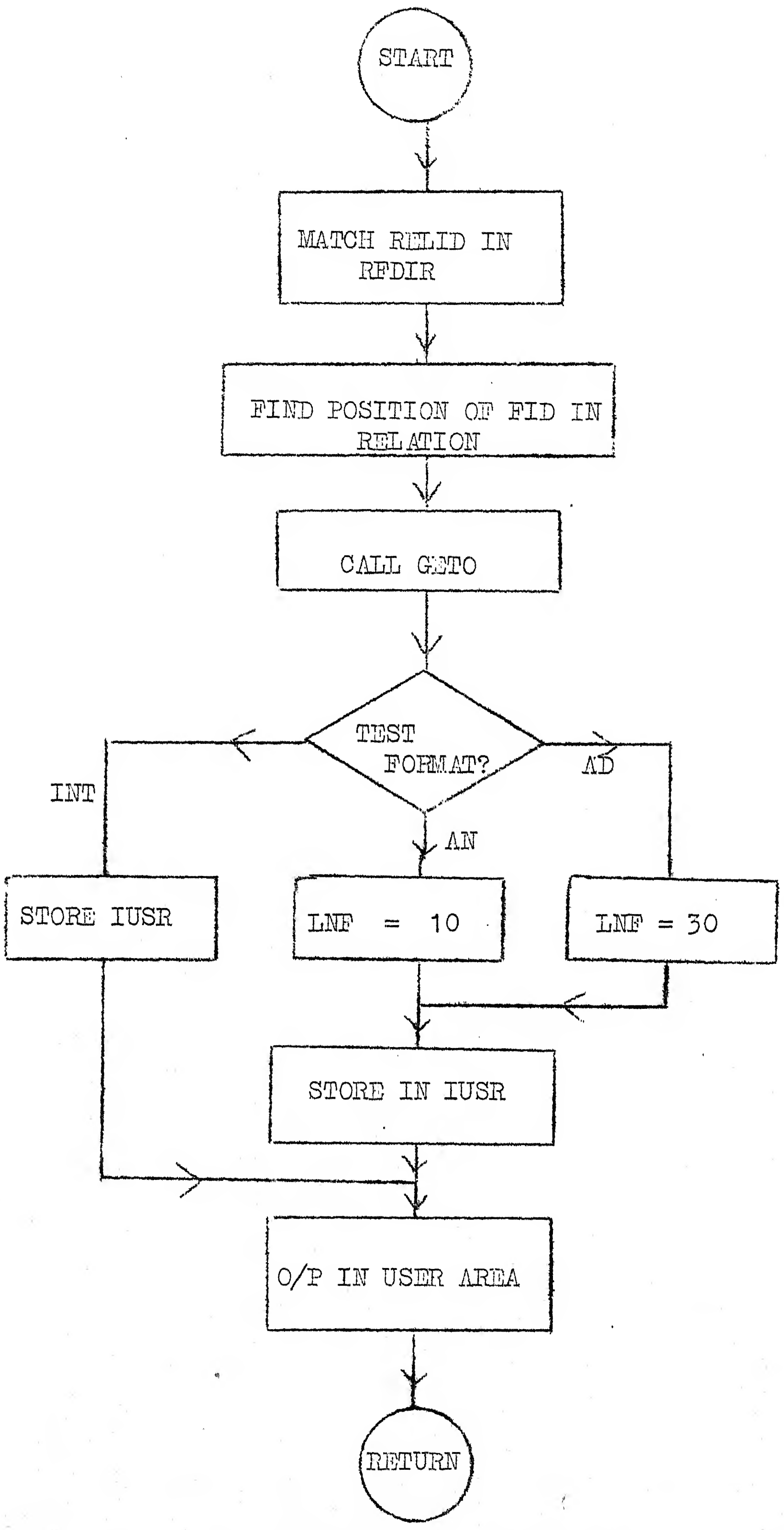


Figure 4.8: Flowchart for Subroutine GETF.

(f) User is given various RELIDs and his Level and asked to continue usage of the system.

4-3.2 <u>Subroutine GETO</u>: This subroutine provides a particular record occurrence or tuple once the relation ID, tuple number and rewind option is specified. Flow chart of subroutine GETO is given in Figure 4.7. The working/functions of the subroutine are:

(a) Locate the Relation ID in RTAB.

(b) Find from the index on which particular device (DNO) - this relation is stored.

(c) On that device find the first relation and its parameters like NI, NA, DA.

(d) Traverse all relations in sequence till desired relation beginning is achieved.

(e) Knowing the displacement of the tuple being IRO traverse till the desired tuple.

(f) Output tuple in user's work area.

(g) Depending on position of rewind flag - rewind the device before returning.

4-3.3 <u>Subroutine GETF</u>: This subroutine is an extension of subroutine GETO, which obtains a particular tuple in a relation. The present subroutine finds out position and format of the desired item/field, whose ID has been specified. The flowchart of the subroutine is given in Figure 4.8 and is self explanatory.

4-3.4 <u>Subroutine GETR</u>: This is a basic subroutine which outputs a complete relation once relation ID has been specified. Same function could be achieved by using GETO till all tuples of a relations are obtained. Here the relation ID is located in RTAB and the index is searched to locate relation on disk. Once allocated, it is outputted.

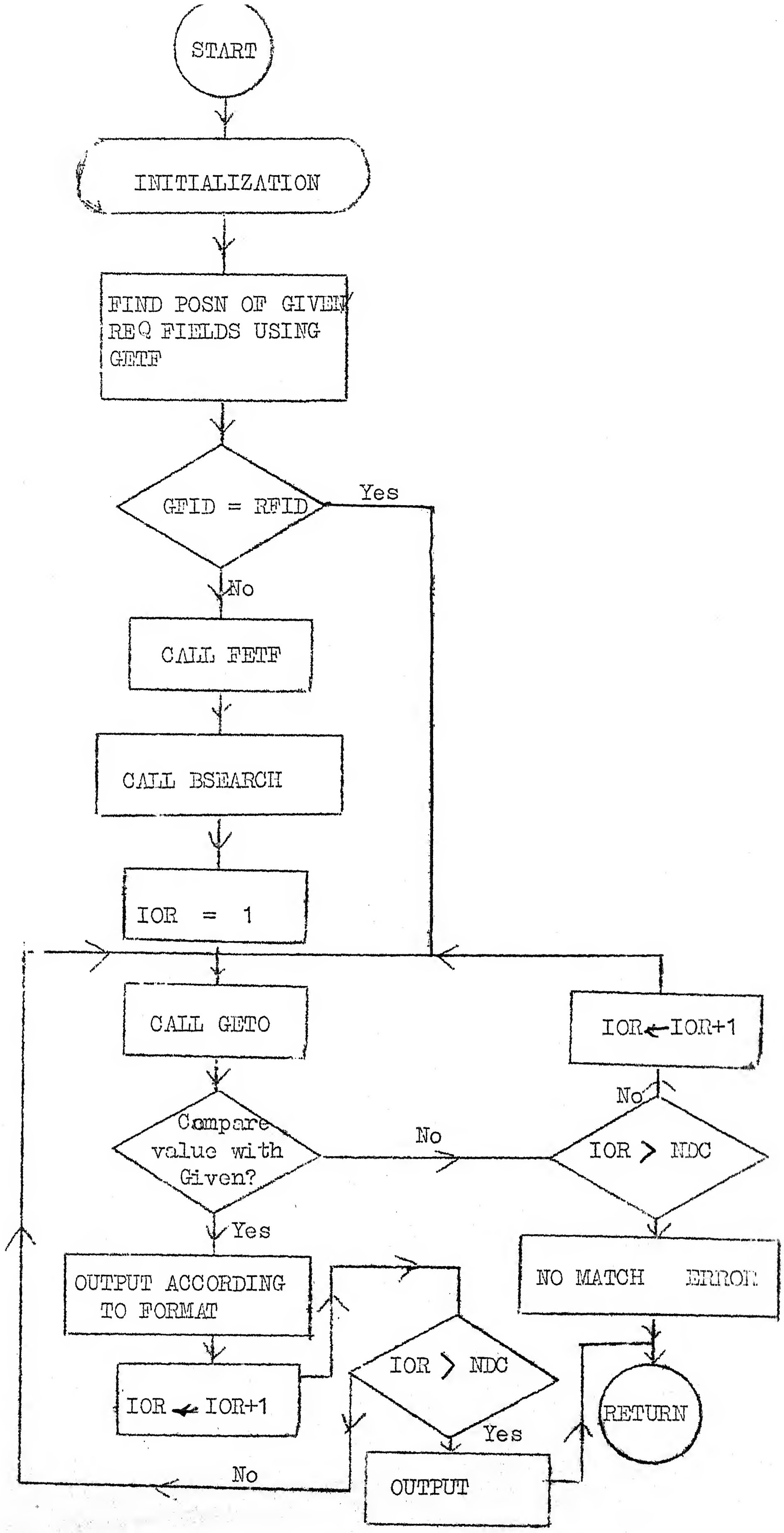


Figure 4.9: Flowchart for Subroutine GETV.

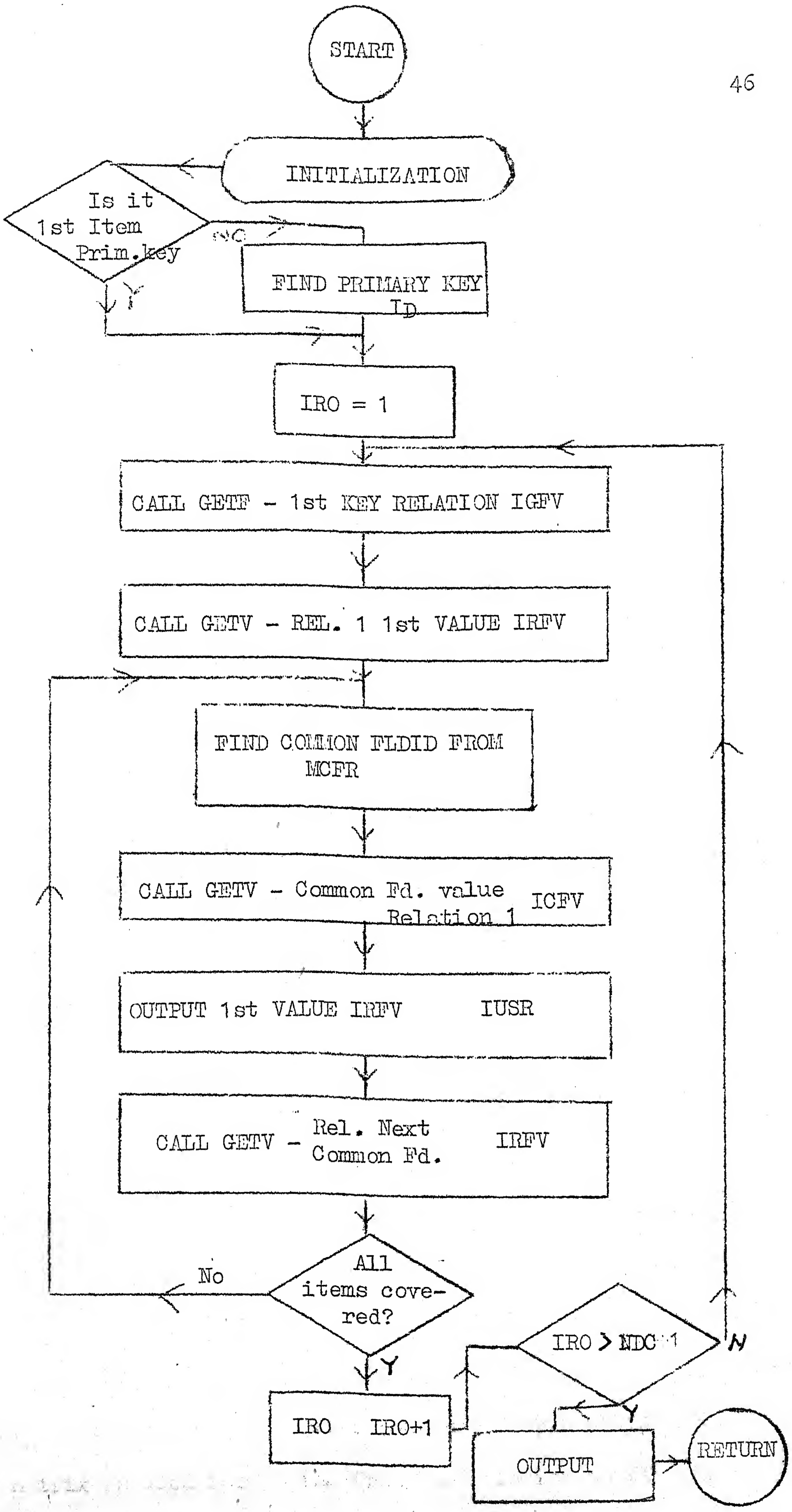
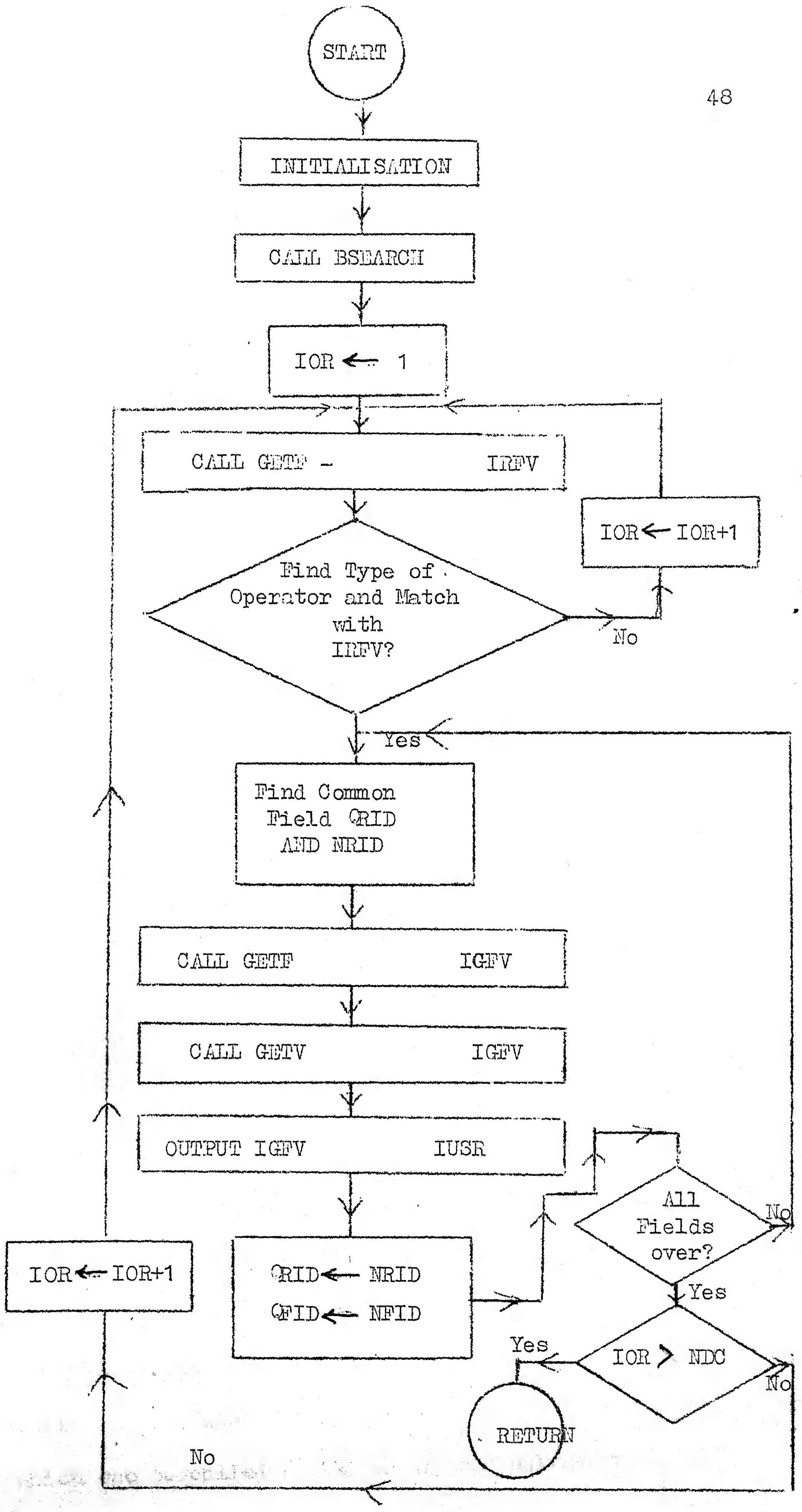

START
INITIALIZATION
Is it 1st Item Prim.key
NO
Y
FIND PRIMARY KEY ID
IRO = 1
CALL GETF - 1st KEY RELATION IGFV
CALL GETV - REL. 1 1st VALUE IRFV
FIND COMMON FLDID FROM MCFR
CALL GETV - Common Fd. value Relation 1 ICFV
OUTPUT 1st VALUE IRFV IUSR
CALL GETV - Rel. Next Common Fd. IRFV
All items covered?
No
Y
IRO IRO+1
IRO > NDC-1
N
Y
OUTPUT
RETURN

Figure 4.10: Subroutine GETU Flowchart.

4-3.5 Subroutine GETV: This is a very useful routine for inverted file type of queries. The subroutine scans a particular given field/item of a relation for the given value. Once the match is obtained in the given field the corresponding value of the required field is outputted according to correct format. Figure 4.9 shows the flow chart for the routine. Extensive use of other routine like GETF, GETO, BSERCH is made in this routine to reduce time in achieving a match. Condition can be set in the subroutine to ensure first value output or all values meeting the requirement. In case of no match of the given value a flag is set and an error message is given to the output.

4-3.6 Subroutine GETU: GETU is one of the important routines which outputs a number of different fields in different relations required by the user based on the primary key of the first relation. This is unqualified GET routine where no conditions are required to be matched. Figure 4-10 gives the flowchart for the subroutines. It extensively uses other routines like GETF, GETV for finding various values. The relation ID's and the corresponding fields are required to be stored in two arrays prior to using the subroutine. This type of tables may be required when a particular report is required to be generated.

4-3.7 Subroutine GETQ: This is the most complex of all the search routines. It outputs a number of fields/items in different relations of target subject to qualification fields satisfying a particular condition. Six relational operators conditions are used as operators for matching values. Extensive use of earlier routines GETQ, GETV is made. Common field matrix is used for moving from one relation to the other. The output is arranged according to the primary key of the qualification field.


START
INITIALISATION
CALL BSEARCH
IOR ← 1
CALL GETF - IRFV
Find Type of Operator and Match with IRFV?
No
IOR ← IOR+1
Yes
Find Common Field QRID AND NRID
CALL GETF IGFV
CALL GETV IGFV
OUTPUT IGFV IUSR
QRID ← NRID
QFID ← NFID
All Fields over?
No
Yes
IOR > NDC
Yes
No
RETURN
No
IOR ← IOR+1

The flow chart for the subroutine is given in Figure 4.11 clearly indicating the variable arrays in which interim values are stored.

4-3.8 <u>Subroutine BSEARCH</u>: This is a general purpose binary search routine used for matching the key values whenever there is a sorted table. The algorithm is based on D.E. Knuth's$^{(26)}$ algorithm. Name of the table and its upper limit are required to be specified along with the key value. The index value is outputted by the routine.

4-3.9 <u>Subroutine IOPT</u>: This is the routine which is also used in IMD analyser and has been covered in ~~Chapter~~ 4-2.5 III.

4-3.10 <u>Subroutine DUMP</u>: To ensure data integrity there may be requirement of taking DUMP at regular intervals or whenever a fatal system error occurs. DUMP is a privilaged subroutine which can be called by DBA or system users with level upto 10 only, to ensure data security. Flowchart for the subroutine is given in Figure 4.12. The system automatically outputs the current contents of all system tables and MCFR. By using subroutine GETRall the relations opened by the user are outputted as DUMP. From DUMP it should be possible for a system analyst/DBA to rectify error.

4-3.11 <u>Subroutine CLOSE</u>: This is the last routine which the user must call, once he has completed use of the data base. This is in a way complimentary to OPEN. The user closing the data base is checked for authorisation. All temporary locations and the users work area are initialised, thus making the system ready for the next user.

4-3.12 <u>Subroutine ADDREL</u>: ADDREL is one of the data manipulation routines which adds a new relation during routine. This is also a privilaged routine which can be called by DBA or system analyst with level less than 10.

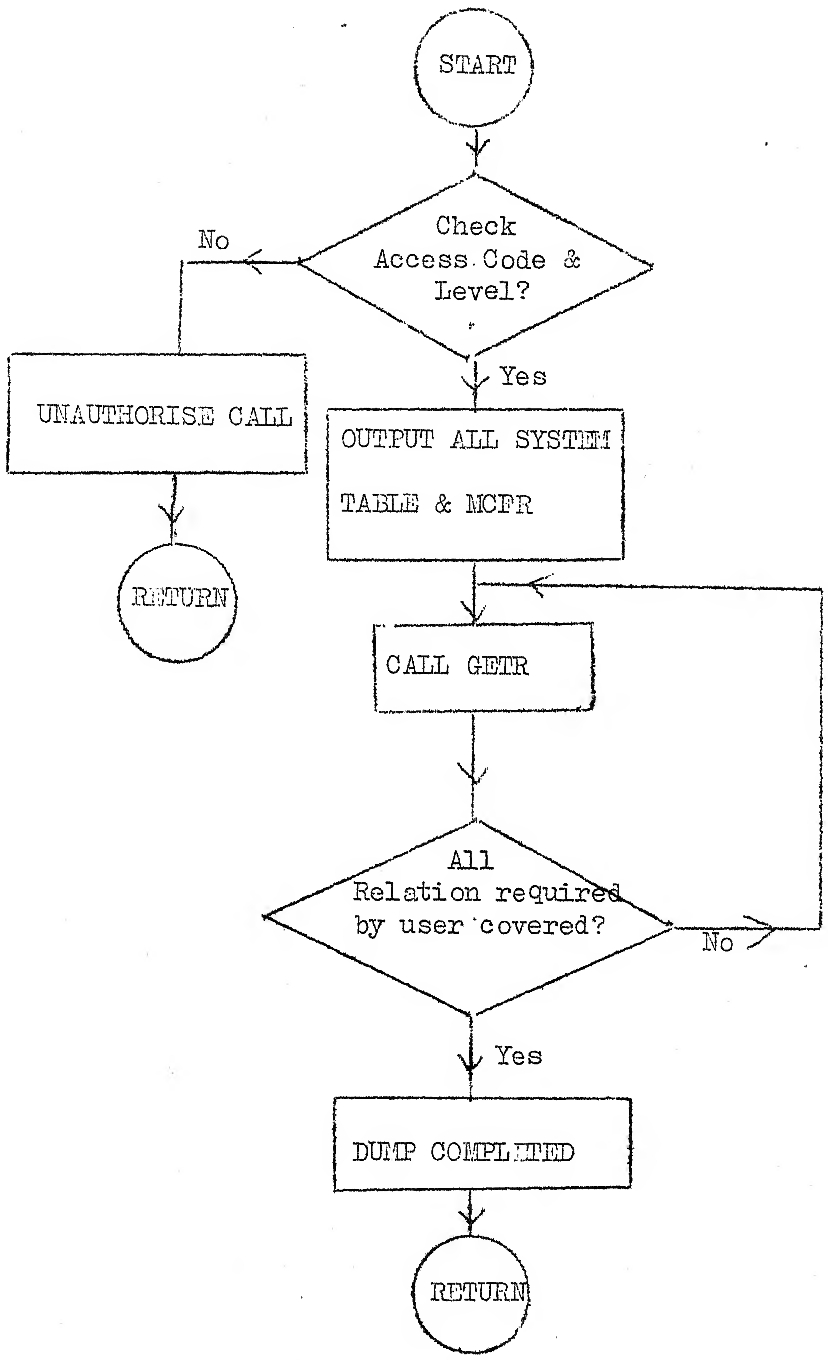

START
Check Access Code & Level?
No
Yes
UNAUTHORISE CALL
RETURN
OUTPUT ALL SYSTEM TABLE & MCFR
CALL GETR
All Relation required by user covered?
No
Yes
DUMP COMPLETED
RETURN

Figure 4.12: Flowchart for Subroutine DUMP.

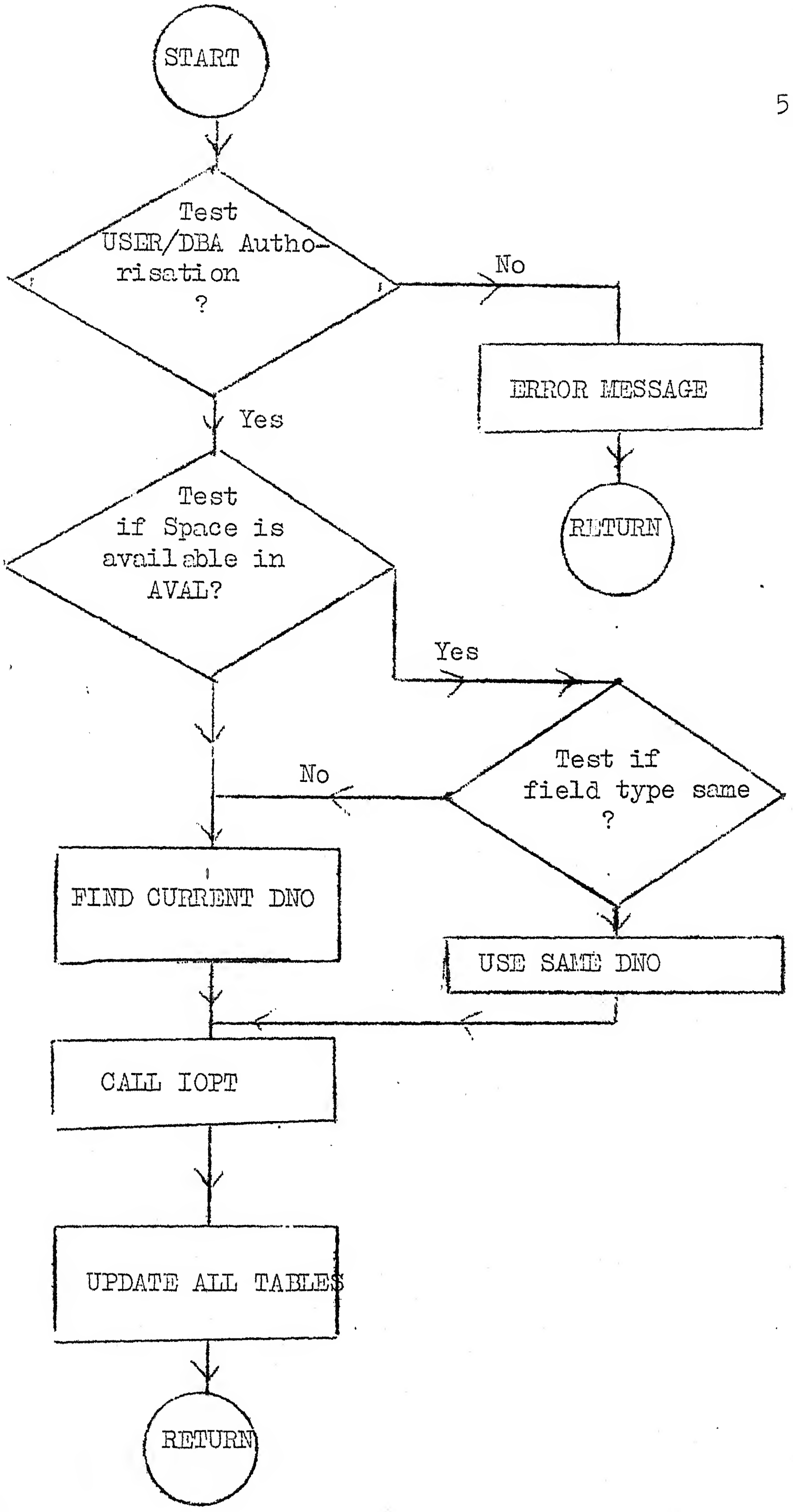


Figure 4.13: Flowchart for Subroutine ADDREL.

The entire data required has to be available in user's work area including name of relation and names of various data item/fields. Flow chart for the subroutine is given in Figure 4.13. The subroutine searches for an earlier deleted relation with similiar or less space requirements. If found it stores in the same DNO, otherwise fresh area is located at the end of all DNO's and the full relation is stored. Relation name is entered into RTAB and new field/item name in FTAB. RFDIR is updated. Once the relation details are [illegible] permanently the system area tables should be rewritten. This is done by subroutine CLOSE.

4-3.13 <u>Subroutine ADDREC</u>: When DMD is originally building the data base 10% extra storage location is kept reserved for any additions at the end of a particular relation. The overflow is worked out as

IOFL = NDC/10 + 1   NDC = Number of tuples in the relation
                         IOFL = Extra storage or overflow area

and the area is stored with zeroes for integer fields and blanks for alphanumeric field. Whenever a tuple is to be added to the relation the availability of overflow area is tested; if full error message is given; else the tuple is placed at the end of the relation. The flow chart for the subroutine is given at Figure 4.14. If overflow area is full in a number of relations, DBA is required to regenerate the system by using DMD module. Once a record is added the RTAB should be rewritten on system area. This is done by subroutine CLOSE.

4-3.14 <u>Subroutine DELTRL</u>: This is also a privileged data manipulation subroutine is allowed to DBA and level below 10. Flowchart for the subroutine is given in Figure 4.15. After checking authorisation for the subroutine - DNO and other parameters of the relation to be deleted are obtained from RTAB index entries. The entire disk space of the relation

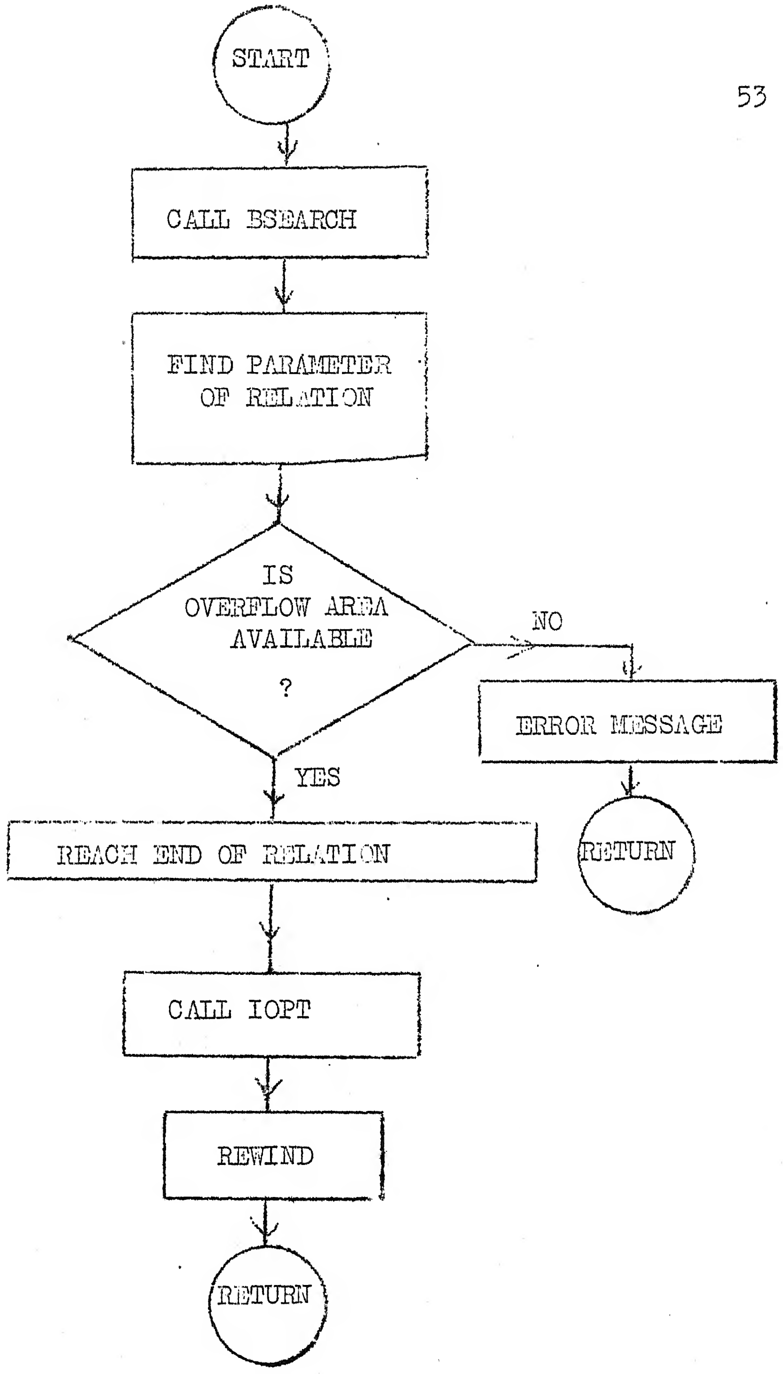


Figure 4.14: Flowchart for Subroutine ADDREC.

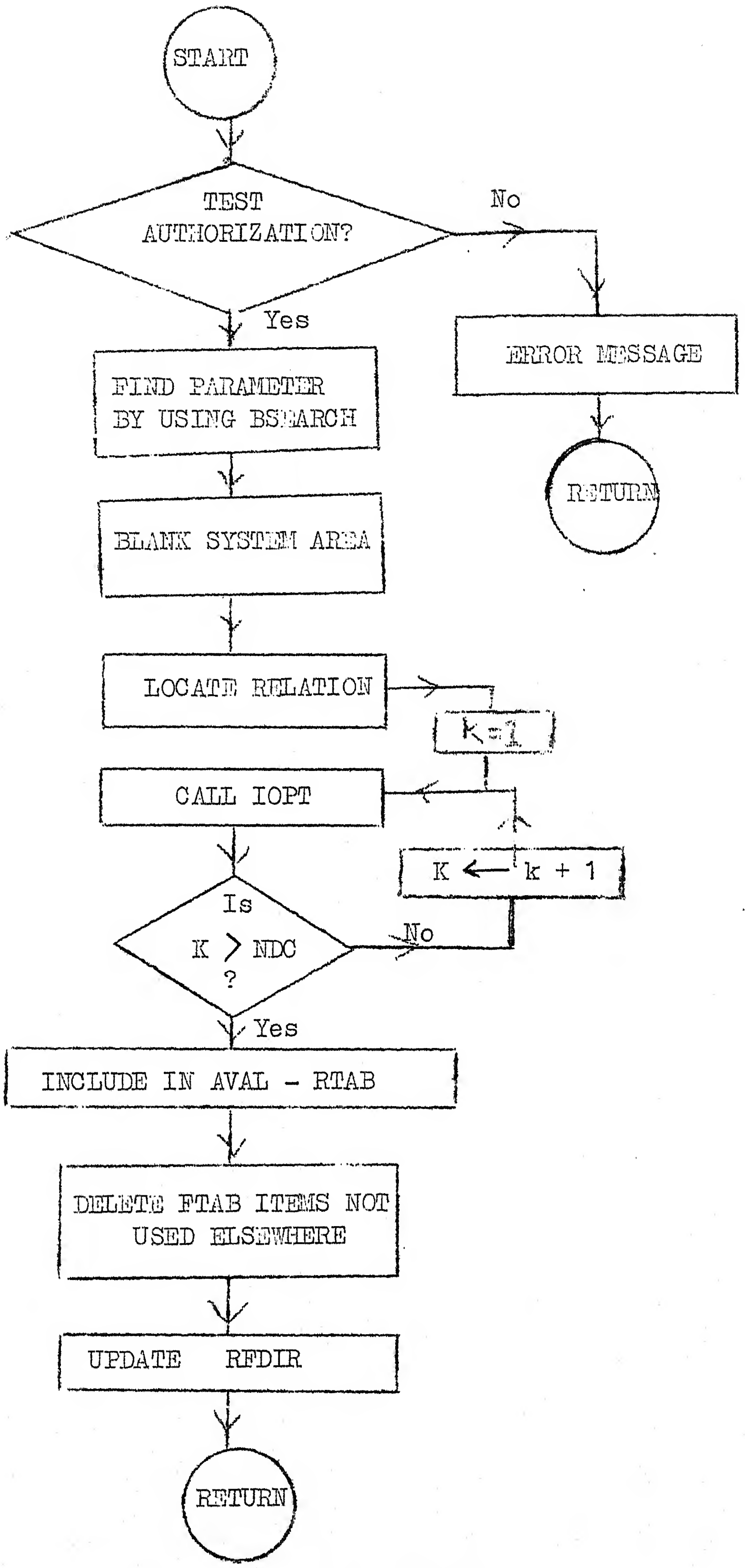


Figure 4.15: Flowchart for Subroutine DELTRL.

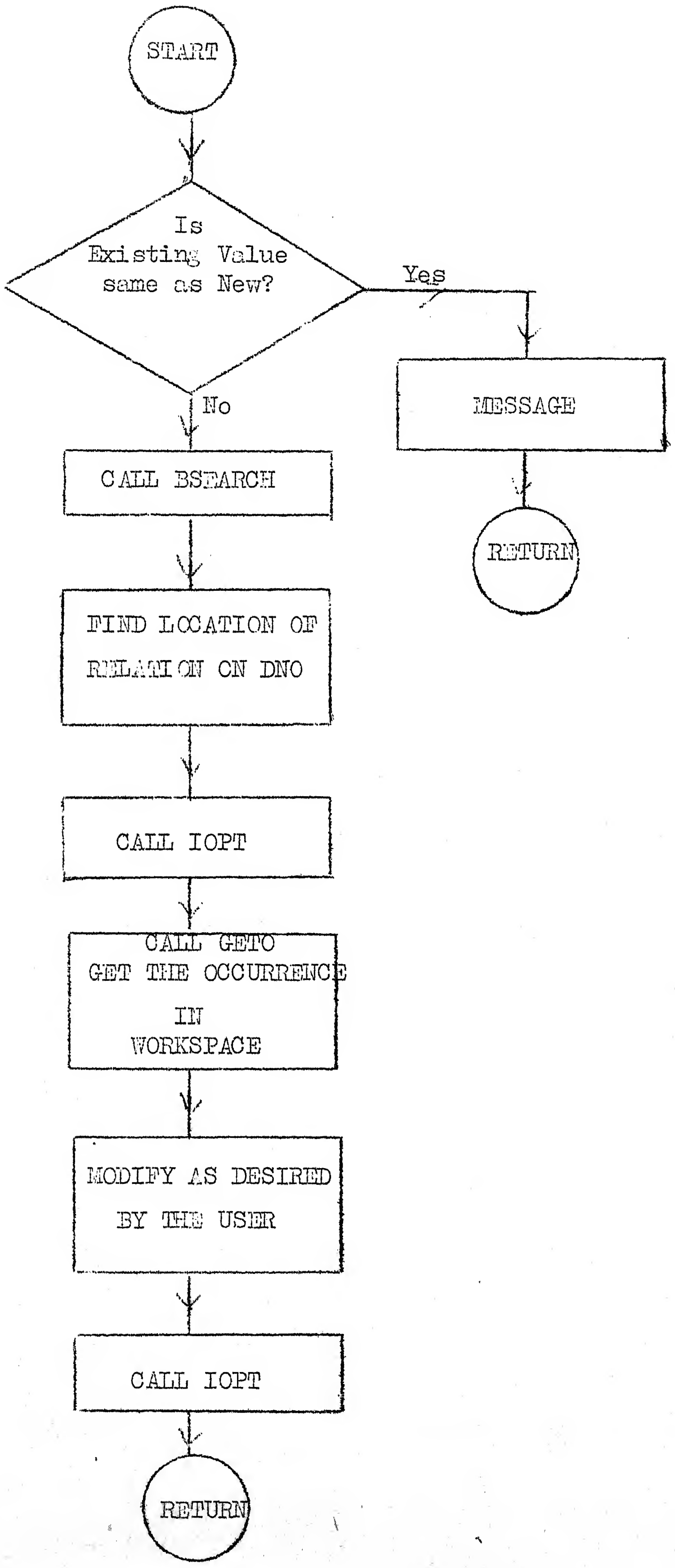

START
Is Existing Value same as New?
Yes
No
MESSAGE
RETURN
CALL BSEARCH
FIND LOCATION OF RELATION ON DNO
CALL IOPT
CALL GETO GET THE OCCURRENCE IN WORKSPACE
MODIFY AS DESIRED BY THE USER
CALL IOPT
RETURN

Figure 4.16 : Flowchart of Subroutine MDYRFC

is blanked. The index is updated to include the deleted area in AVAL list. RTAB is accordingly amended. FTAB is brought uptodate by deleting all items which are not referenced/required by any other relation in the data base. Similarly RFDIR is corrected to incorporate changes in the data base. After all the modifications are incorporated, the amended system tables should be rewritten in system area for use, using CLOSE.

4-3.15 Subroutine MDYREC: This subroutine can modify any tuple or field in the data base. If the record is to be deleted it is replaced by zeroes and blanks in their appropriate positions integer field replaced by zeroes and alphanumeric by blanks. The flowchart for the subroutine is given in Figure 4.16. The existing value is checked with the new value and if both are identical, no further action is taken. The occurrence is to be modified is brought into buffer and modified, as desired by the user. The same is rewritten using IOPT. In all data manipulation routines, subroutine GETO with non-rewind option is used to reach upto the desired occurrence for modification. Once the action is complete the appropriate DNO is rewound.

CHAPTER V

SYSTEM PERFORMANCE EVALUATION

5.1 General

An experimental relational DBMS as an extension to FORTRAN has been implemented on TDC-316 at Indian Institute of Technology, Kanpur. As indicated in Chapter IV, the implementation has been achieved in two independent modules - the DMD Analyser and DSL including DML and utility routines. The program listing has been attached at Appendix 'B'. For system performance evaluation live data from two different defence laboratories has been used avoiding any classified information. The system has been tested for 11 different relations having variable record lengths, formats and record occurrences. In the absence of any existing working system, the system performance has been discussed in absolute terms; rather than any comparative evaluation.

5.2 Sample Outputs

5-2.1 DMD Analyser: A sample output of DMD analyser module on the teletype is shown in Table 5.1. The building of the relational data base is done by the data base administrator so DBA code has to be matched. Once the authentication is completed the DMD Analyser asks various questions regarding the layout, formats, length of each relation and guides the DBA to proper generation of the system. A hard copy of the relation generated is kept on Device 6 (Printer in this case) for reference of the DBA. Once all the relations are completed and IEND card is sensed; the various system data structures like RTAB, FTAB and RFDIR are displayed for verification by the DBA. The system tables are stored on the system

```
MAY I HAVE YOUR IDENTIFICATION PLEASE?  WELCOME SIR! DBA AUTHORISED CODE --
DB 1 BUILDING A RELATIONAL DATA BASE FOR DRDO LABS
--------------------------------------------------------------------------

 PLEASE GIVE RELATION DEFINITION AS RELN NAME AC?
RELN PROJ AB
FIELD DEFINITION AS FLDS NAME IN OR AN OR AD
AT END GIVE NO OF DATA CARDS AS DATA XX 4 BLANKS
FLDS CODE IN
FLDS CRNO IN
FLDS TYPE IN
FLDS NAME AN
FLDS LAB AN
FLDS PPNO AN
DATA 10
PLACE  10 DATA CARDS IN CARD READER!
 IS IT ALL? GIVE IEND CARD ELSE GIVE RELN CARD!
RELN PROG AB
FLDS CODE IN
FLDS POSN AD
DATA 10
PLACE 10 DATA CARDS IN CARD READER!
 IS IT ALL?  GIVE IEND CARD ELSE GIVE RELN CARD!
IEND
         ALL RELATION IN DATA BASE
         -------------------------

                 CONTENTS OF RTAB
        IPROJ   4    1AB      10      3     3      0
        2PROG   4   13AB      10      1     0      1
            NUMBER OF ENTRIES IN TABLE     --         2

                 CONTENTS OF FTAB
            FLDID       NAME   FORMAT
                1       CODE        1
                2       CRNO        1
                3       TYPE        1
                4       NAME        2
                5       LAB         2
                6       PRNO        2
                7       POSN        3
             NUMBER OF ENTRIES IN TABLE    --         7

                  CONTENTS OF RFDIR
    RELID            FLDID            FORMAT
        1                1                 1
        1                2                 1
        1                3                 1
        1                4                 2
        1                5                 2
        1                6                 2
        2                1                 1
        2                7                 3
             NUMBER OF ENTRIES IN TABLE    --         8
--------------------------------------------------------------------------
                 DMD ANALYSER COMPLETED
--------------------------------------------------------------------------
```

Table 5.1: DMD Analyser Sample OUTPUT on TTY.

```
          USER ! IDENTIFY ACCES CODE IN A2 AND LEVEL IN I2?
AB01
          HOW MANY RELATIONS YOU WOULD BE USING?
08
          RELATION NAME IN 2A2 FORMAT PLEASE?
PROJPROGDATSFCODFUNDPAGNECENPREC
          AB        AUTHORISED USER  ---- LEVEL, -----1

                HERE ARE RELATIONS ID

          RELID           NAME
            1             PROJ
            2             DATS
            3             FCOD
            4             PROG
            5             FUND
            6             PAGN
            7             PREC
            8             ECBN
            9             PRTY
           10             MANU
           11             RKCD
          ------------------------------
               OPEN TEST COMPLETED
          ------------------------------

          READING FROM DEVICE --4 WRITING ON -- 1
11              1IMPROVE FA RADAR150LRDE BANGALORE     3L-PX-67/LRD-32
          TESTING WRITE PROTECTION FOR USER CODE -- 51
SORRY !  YOU ARE NOT ALLOWED TO WRITE IN DB !

-------------------------------------
          IOPT TEST COMPLETED
-------------------------------------
```

Table 5.2: Sample Outputs OPENAND IOPT

```
       GIVE LOCATION OF RELID -- 2 IN RTAB ?
RELID    2  MATCHED IN RTAB AT -- 2
---------------------------------------------
       BSEARCH TEST COMPLETED               -
---------------------------------------------

      GET REC OCCURRENCE -- 4 OF RELID -- 1 REWND 1
 4 0  2 CS ASS FOR FMOW MADLRDE BANGALORE   RD-P1-69/LRD-84
      GET REG OCCURRENCE -- 6 OF RELID -- 3 REWND 0

1001 GPA FOR WS C42                         -
---------------------------------------------
           GETO TEST COMPLETED              -
---------------------------------------------

      GET 5 FLDID OF 1 RELID 6 REC OCCURRENCE ?

GETF OF -- RID - 1 FID 5 FROSN 14 FOT  2
GPA FOR WS C42

---------------------------------------------
           GETF TEST COMPLETED
---------------------------------------------
```

Table 5.3: Sample Outputs BSERCH, GETO AND GETF

device 16. Thus once DMD Analyser work is over the module can be thrown out of the system.

5-2.2 <u>DSL Module</u>: This is the main module of DBMS for users application and consists of main program and 15 different routines. The outputs of the important routines are given in Table 5.2 onwards.

(a) <u>OPEN/IOPT</u>: As shown in Table 5.2 after verification of the user code and level the module questions the user on number of relations required, as well as their names. User level is verified for Read/Write option. Once the user's identity is confirmed, the module generates the common matrix MCFR for use by the system. IOPT output for two cases is also shown.

(b) <u>BSEARCH/GETO/GETF</u>: The sample outputs of the three routines are also given in Table 5.3. Once relation ID is specified the routine BSEARCH locates the position of the relation in RTAB. GETO is the most commonly used routine, which outputs a particular record occurrence once Relation ID and occurrence position with respect to beginning of relation is specified. In the sample output first relation 4th occurrence with Rewind and third relation 6th occurrence without rewind were required. GETF provides a particular field (item) value one relation ID, Field ID and record occurrence is specified. In this case the case third field from first relation and 6th occurence was required.

```
          GET FLDID == 6 FROM RELID 1 WHERE VALUE FID == 5
                DLRL HYDERABAD

          SL=PX=66/DLR=18
          RD=P1=67/DLR=24
          SA=PX=67/DLR=26
          SN=P1=68/DLR=28
          SN=P1=68/DLR=89

==================================================================
                      GETV TEST COMPLETED
==================================================================

          GET WHOLE RELATION WITH ID == 1

0011 000   1IMPROVE FA RADAR1501LRDE BANGALORE    SL=PX=67/LRD=32

00020000   02FM/CWPDOPPLER RADAR LRDE BANGALORE   RD=P1=68/LRD=76=2

0003       2DIG RADAR DISPLAY EQLRDE BANGALORE    RD=P1=72/LRD=83 :1

0004       2 CS ASS FOR FMCW RADLRDE BANGALORE    RD=P1=69=/LRD=84

0005       2CCBS INT CIR OF RADALRDE BANGALORE    RD=PX=70/LRD=91

1001       1GPA FOR WS C42     DLRL HYDERABAD     SL=PX=66/DLR=18

1002       2 DF AREAIL HF BAND DLRL HYDERABAD     RD=P1=67/DLR=24

1003       4HF LOG  FERIODIC ANT DLRL HYDERABAD   SA=PX=67/DLR=26

1004       8BB HIGH POWER TX  DLRL HYDERABAD      SN=P1=68/DLR=28

1005       8HF MONOCONE AREIL DLRL HYDERABAD      SN=P1=68/DLR=29
           ===================================
              GETR TEST COMPLETED
           ===================================
```

Table 5.4: Sample Outputs GETU AND GETR

```
                FROM RELATIONS ID'S --  1   2   3
                GET VALUES FLDID'S  --  3   1  10

RELID        FLDID        IRO          FORMAT          VALUE
  1            3           1             1               1
  2            1           1             1               1
  3           10           1             2         TOTE DISPLAY RADAR
  1            3           2             1               2
  2            1           2             1               2
  3           10           2             2         FM/CWPDOPPLER RADAR
  1            3           3             1               2
  2            1           3             1               3
  3           10           3             2         DIG RADAR DISPLAY EQ
  1            3           4             1               2
  2            1           4             1               4
  3           10           4             2         CS ASS FOR FMCW RAD
  1            3           5             1               2
  2            1           5             1               5
  3           10           5             2         CCBS INT CIR OF RADA
  1            3           6             1               1
  2            1           6             1               1001
  3           10           6             2         GPA FOR WS C42
  1            3           7             1               2
  2            1           7             1               1002
  3           10           7             2         DF AREAIL HF BAND
  1            3           8             1               4
  2            1           8             1               1003
  3           10           8             2         HF LOG PERIODIC ANT
  1            3           9             1               3
  2            1           9             1               1004
  3           10           9             2         HIGH POWER TRANS
  1            3          10             1               3
  2            1          10             1               1005
  3           10          10             2         HF MONOCONE AREIL

------------------------------------------------------------------
                   GETU TEST COMPLETED
------------------------------------------------------------------
```

Table 5.5: Sample Output GETU

```
        GIVEN RELID --1 FLDID -- 1 VALUE -- 1001
        GET VALUES OF FLDIDS -- 3       1       10
        FROM RELATIONS IDS   -- 1       2        3

GETQ    OP -- RID     1   FID    13 [illegible] IOR    6   IFOT   1
                           1
GETQ    OP--  RID     2   FID     1     IOR    6   IFOT   1
                         1001
GETQ OP-- RID 3 FID  10 IOR   6 IFOT 2
GETQ    OP -  RID     3   FID    10    IOR    6   IFOT   2
        GPA FOR WS C42
---------------------------------------------------------------------
                   GETQ-TEST COMPLETED
---------------------------------------------------------------------
```

Table 5.6: Sample Output GETQ

```
     RECORD TO BE ADDED IN RELID -- 2 FOLLOWS

10101    101010101    1010

TESTING IOR--11 IN RELID -- 2

10101    101010101   1010

----------------------------------------------
        ADDREC TEST COMPLETED
----------------------------------------------

     MODIFY RELID--1 FLDID--5 IOR-- 3 AS BELOW

            DLRL HYDERABAD

3    0    2DIG RADAR DISPLAY EQLRDE BANGALORE     RE-P1-72/LRD-83
30   0    2DIG RADAR DISPLAY EQDLRL HYDERABAD     RD-P1-72/LRD-83

           TESTING THE NEW VALUE USING GETF

                 DLRL HYDERABAD

---------------------------------------------
       MDYREC TEST COMPLETED
---------------------------------------------
```

Table 5.7: Sample Outputs ADDREC and MDYREC

(c) GETV/GETR: Table 5.4 shows the output of GETV routine. This is a very powerful routine which matches the desired relation for a particular value and outputs a corresponding field. In this case name of laboratory was given as DLRL Hyderabad and project numbers were required from Relation ID 1 'DLRL HYDERABAD'. GETR outputs a complete relation as it exists in the secondary storage. Relation with ID as 1 has been chosen in the output.

(d) GETU: GETU command outputs the desired fields from different relations. In Table 5.5, third field of first relation, first field of 2nd relation and 10th field of 3rd relation have been outputted based on first mentioned field as primary key.

(e) GETQ: This is the most complex routine which outputs the desired fields from different relations once the condition of a particular field is satisfied. In Table 5.6 the same fields of GETU have been outputted such that their Project ID is 1001.

(f) ADDREC/MDYREC: These are the privileged DML routines which add a particular record at the end of the desired relation or modify a particular field in the specified occurrence. The outputs of these routines are given in Table 5.7.

(g) ADDREL/DELTRL: Table 5.8 gives the output when a new relation is to be added at the end of all relations and also if the relation is required to be deleted. These are privilaged routines allowed upto user level 10. It may be noted that system tables are extensively amended by both routines. The modified tables are stored on system area using CLOSE.

(h) DUMP/CLOSE: These are the utility routines whose sample teletype outputs are given as Table 5.8. The routine DUMP outputs all system tables and the contents of all relations used by the users. This output is given on printer. The CLOSE routines initialises all the buffers, the system devices and restores the system area for subsequent use. This routine is complementry to OPEN.

5.3 Evaluation of System Developed

Some of the important parameters for evaluation of any DBMS are integrity, security, time for response and recovery facilities. In minicomputer environment the memory used is also an important consideration. The performance of the system has been discussed below for their vital parameters.

5-3.1 Integerity and Accuracy: The system, being FORTRAN based, has inherent integerity as any error in the type of data is automatically detected. In the routines developed additional hard copy of entire transactions is kept on Device 6 which could be regularly monitored to detect any malfunctinning. Taking DUMP may also help in detecting any errors as such a seperate DUMP routine has been provided. In all the runs, a very high degree of accuracy in reproducing data was observed. The system is as accurate as any FORTRAN system.

5-3.2 Security: In the system developed each relation has been provided with a two characters security code, which has to be matched by every user before he can access the data base. In addition each user is also allotted a level between 00 and 99 to ascertain his priority. Restriction on use of any routine by a particular level can be enforced. In the present system all users having level greater than 50 can only READ from data base. Users having level less than 10 can use the privilaged routines ADDREC, MDYREC, ADDREL and DELTRL. In case of any unauthorised use the user is stopped and a warning is given on the teletype. Thus security and privacy have been ensured to a large extent.

5-3.3 Time for Response:

(a) DMD Analyser: The system was used to generate 11 relation having various record occurrences upto 15. The entire data model definitions was completed within 5 minutes.

(b) DSL Routine: In the absence of in built clock, the time of response was measured using stop watch. The limiting factor in most of the queries was the teletype speed. Normal queries regarding a single entity were answered within 5-10 seconds. When the entire relation is to be matched and different fields are to be outputted as in GETU it takes longer time. Similarly in adding records/modifying record the time is around 5 seconds. Routine like DUMP, ADDREL take 3-5 minutes depending on Data base contents. In the type of usage envisaged the response time is fairly satisfactory.

5-3.4 Memory Utilised: TDC-316 configuration at IIT Kanpur has available memory of 28K words (16 bits). In FORTRAN, the run time routine including KDM and IOCS take around 10.2K, thus leaving only approximately 18K words for the system development. DMD Analyser designed took approximately 11K word of memory. Since the module is to work independently so effort was made to reduce memory usage. The DSL and DML routines including utility routines together occupy about 16K word including 2K word user buffer area. Considering the features supported this is fairly economical in memory usage. However, the memory used could be reduced by using BAL-316.

5.4 Constraints of the System Developed

Some of the constraints, which are normally present in any FORTRAN programming are also present in the software developed. Considering the time and memory available the following constraints are specified.

(a) A maximum of 20 relations can be generated at one time.

(b) Maximum No. of fields (distinct) can be upto 40.

(c) Maximum entries in RFDIR should not exceed 50.

(d) Maximum length of any records cannot exceed 80 columns.

(e) Only 10% extra space is catered for adding new records. In case of excess the DMD has to be redone.

(f) The input cards are fixed formatted and not extra blanks are tolerated.

Most of the constraints are dependent upon the dimensions and as such can be easily modified, if required, in any typical application. As it is the system is fully workable for any scientific or business mini data base.

CHAPTER VI

SCOPE FOR FURTHER WORK AND CONCLUSION

6.1 Summary

For implementation of data base management system on TDC-316, the relational approach was adopted, primarily for ease of handling normalised relation. FORTRAN language was chosen for system development because of its popularity and the ~~inherent~~ availability of a good subroutine structure. A simple fixed format data model definition language was used to provide ease in building the data base for the data base administrator. Only three basic types of data structures (integer, 20 characters alphanumeric and 60 characters alphanumeric) were included in the design, as these would meet most of the requirements of data processing. The various facilities for retrieving, adding, deleting and modifying data have been provided by a set of 15 data sublanguage and data manipulation language routines.

6.2 Difficulties in Implementation

One of the major factors in limiting the design of DBMS has been the limited run time memory available. (Only 18K in FORTRAN on TDC-316 at IIT Kanpur). Every effort has been made to make the system as general as possible within the available memory. The other difficulty has been the reliability of TDC-316 computer system, especially the peripheral devices. Like printer, disk and teletype were often defective, resulting in considerable loss of time in proving system software. In case the present experimental system has to be made commercial; the reliability of the TDC-316 will have to be improved.

5.3 Recommendations for Further Work

The under mentioned work is suggested to follow up the successful impelementation of the relational DBMS:

(a) Data Validation: In the present system data validation is done by FORTRAN run time routine which gives error message on detecting wrong type of data. However, there is requirement for some general purpose data validation routines to improve the system integrity and accuracy.

(b) Interactive Working: DMD module has been made completly interactive, whereby DBA is guided in building the data base. Dialogue on teletype has been incorporated for opening of the data base. It is recommended that a general main program using all these routines be made whereby an ordinary user is questioned on his requirements and the appropriate routine is chosen for meeting his needs.

(c) Disk Operating System: KDM-316 version I with some modifications has been incorporated as data base control system. ECIL Hyderabad has recently announced their Basic Disk Operating System. Similarly a more efficient DOS is being developed by TIFR. It is suggested that efforts should be made to integerate these disk operating systems with the present software. This will cut down the response time and improve the efficiency in disk utilisation.

(d) <u>Unformatted Transfer of Data</u>: TDC-FORTRAN does provide a facility for transfer of unformatted data. In such case the transfer of data between various devices/buffers will become more efficient. The feasibility of incorporating the same may be studied.

(e) <u>General Purpose Software</u>: The existing software has certain machine dependent feature to make it more efficient for the TDC-316. The software can be modified to make it general purpose and thus transportable for any machine.

(f) <u>3NF Relations</u>: In the present implementation, it is assumed that the relation is in the 3rd Normal Form. Some algorithm can be developed to optimise the number of normalised relations in any data base, once the raw data is made available.

(g) <u>Other Languages</u>: The relational approach has been implemented both in FORTRAN and BAL-316(26,27,28). It may be worthwhile designing a hybrid system based on FORTRAN and BRASS (TIFR version). Such a system will have the advantages of both an assembly language and a higher level language.

(h) <u>Other Approaches</u>: CODASYL and HIERARCHICAL approaches may also be implemented in FORTRAN, so that a comparative evaluation of all the three approaches can be attempted.

6.4 Conclusion

The implementation of a relational data base management system as an extension to FORTRAN has been successfully completed. The system software developed, though experimental, can be implemented in its present form in any organisation having TDC-316 computer system.

REFERENCES and BIBLIOGRAPHY

1. Martin, J., "Principles of Data Base Management", Prentice-Hall, Inc., Englewood Cliffs, N.J., 1976, Chapter 1.

2. Engles, R.W., "A Tutorial on Data Base Organization", Annual Review in Automatic Programming, Vol. 7, Part 1 (edited by Halpen and McGee), Pregamon Press, July 1972.

3. Fry, J.P., and Sibley, E.H., "Evolution of Data Base Management Systems", Computing Survey, Vol. 8, No. 1, March 1976, pp. 7-42.

4. Martin, J., "Computer Data Base Organization", Prentice-Hall, Inc., Englewood Cliffs, N.J., 1975.

5. Date, C.J., "An Introduction to Data Base Systems", Addison-Wesley Publishing Company Inc., Phillipines, 1975.

6. Haseman, W.D., et. al., "A Partial Implementation of CODASYL DBTG Report as an Extension to FORTRAN," Management Datamatic, Vol. 4, No. 5, October 1975, pp. 75-97.

7. Tsichritzis, D.C., and F.N. Lochovsky, "Hierarchical Data Base Management : A Survey", Computing Survey, Vol. 8, No. 1, March 1976, pp. 105-123.

8. "CODASYL DATA BASE TASK GROUP", April 1971, Report ACM, New York.

9. Codd, E.F., "A Relational Model of Data for Large Shared Data Banks", Communication of ACM, Vol. 13, No. 6, June 1970.

10. Chamberlin, Donald, D., "Relational Data Base Management Systems", Computing Survey, Vol. 8, No. 1, March 1976, pp. 43-66.

11. Codd, E.F., "A Data Base Sub Language Founded on Relational Calculus", Proc. 1971 ACM-SIGFIDET Workshop on Data Description, Access and Control, Nov. 1971, ACM, New York, 1971, pp. 35-68.

12. Stewert, J., and J. Goldman, "The Relational Data Management System : a Perspective", Proc. ACM-SIGMOD Workshop on Data Description Access and Control, May 1974, ACM, New York (1974), pp- 295-320.

13. Bracchi, G., A. Fedeli, and P. Padlini, "A Language for a Relational Data Base," Sixth Annual Princeton Conf. on Information Sciences and Systems, March 1972, Princeton University, N.J., 1972.

14. Fehder, P.L., "The Representation Independent Language", IBM Research Report RJ 1121 and RJ 1251, San Jose, Calif., Nov. 1972 and July 1973 respectively.

15. Astrahan, M.H., and D.D. Chamberlin, "Implementation of a Structured English Query Language", Communication of ACM, Vol. 18, No. 10, (October 1975), pp. 580-588.

16. Todd, S.J.P., "Peterlee Relational Test Vehicle PRTV, a Technical Overview", IBM Scientific Centre Report JKSC 0075, Peterlee, England, July 1975.

17. Whitney, V.K.M., "RDMS a Relation Data Management System", Proc. Fourth International Symposium on Computer and Information Sciences, (COINS IV) Dec. 1972, Plenum Press, New York, 1972.

18. Boyce, R.F., D.D. Chamberlin, W.F. King, and M.M. Hammer, "Specifying Querries as Relational Expressions : The SQUARE Data Sub Language", Communication ACM, Vol. 18, No. 11 (Nov. 1975), pp. 621-628.

19. Chamberlin, D.D., and R.F. Boyce, "SEQUEL : A Structured English Query Language", Proc. ACM-SIGMOD Workshop on Data Description, Access and Control, May 1974, ACM, New York, 1974, pp. 249-264.

20. Copland, G.P., and S.Y.W. Su, "A High Level Data Sub Language for Context Addressed Segment Sequential Memory", Proc. ACM-SIGMOD Workshop, May 1974, ACM, New York, pp. 265-276.

21. McDonald, N., and M. Stonebraker, "CUPID : The Friendly Query Language", Proc. ACM Pacific 75 Regional Conf., April 1975, ACM, New York, 1975, pp. 132-139.

22. Zloof, M.M., "Query by Example : The Invocation and Definition of Tables and Forms", Proc. International Conf. on Very Large Data Bases, Sept. 1975, ACM, New York, 1975, pp.1-24.

23. Codd, E.F., "Recent Investigations in Relational Data Base Systems", Information Processing 74, Proc. IFIP Congress, August 1974, Vol. 5, North-Holland Publ. Co., Amsterdam, The Netherlands, 1974, pp. 1017-1021.

24. Sharman, G.C.H., "A New Model of Relational Data Base and High Level Languages", Technical Report TR12.136, IBM Huskey Park Laboratory, England, Feb., 1975.

25. ECIL-Hyderabad, "TDC-316 FORTRAN", 1976.

26. Knuth, D.E., "Art of Computer Programming Vol. III - Sorting and Searching", Addison-Wesley, 1975.

27. Sharma, R.K. (Major), "A Personnel Data Base Retrieval System", M.Tech. Thesis, Computer Science Program, IIT-Kanpur, July 1977.

28. Mukherjee, A.K., "Data Validation and Normalizing for a Personnel Data Base Built on the Relational Approach", M.Tech. Thesis, Computer Science Program, IIT-Kanpur, July 1977.

29. Ghanekar, D.K., "A Relational Implementation of Data Base System", M.Tech. Thesis, Computer Science Program, IIT-Kanpur, July 1977.

APPENDIX 'A'

PATCHES IN RUN TIME CONTROL ROUTINES FOR TDC FORTRAN

Changes incorporated in FC Routines

```
*043054 ---> 00350   JMS PC, GETDST
     56 ---> 042502

*043100 ---> 003050  No open : JMS, PC, GETBOT
    102 ---> 042546
    104 ---> 100
    106 ---> 100

*042546 ---> 050210 GETBOT : BCMP R2,# 13
     50 ---> 15
     52 ---> 011410 BRGT # 20
     54 ---> 121050 TSR #62, @16464
     56 ---> 62                                  D1
     60 ---> 16464
     62 ---> 121050 TSR # 1, @ 16466
     64 ---> 1
     66 ---> 16466
     70 ---> 250  JMP RETRN
     72 ---> 43002
     74 ---> 050210 BCMP R2,# 14
     76 ---> 16

*042600 ---> 011407 BRGT, +20
    602 ---> 121050 TSR # 144, @ 16500
    604 ---> 144
    606 ---> 16500                              D2
    610 ---> 121050 TSR # 1, @ 16502
    612 ---> 1
    614 ---> 16502
    616 ---> 012071 BRN RETRN
    620 ---> 012050 BRN CLOSE

*042742 ---> 050210 CLOSE : BCMP R2, # 15
    744 ---> 17                                 D3
    746 ---> 011407 BRGT.+20
    750 ---> 011407 TSR # 226, @ 16514
    752 ---> 226
    754 ---> 16514
    756 ---> 121050 TSR # 1, @ 16516
    760 ---> 1
    762 ---> 16516
    764 ---> 012006 BRN.+16
    766 ---> 121050 TSR # 36, @ 165530
```

```
*42770 ——> 36
  2772 ——> 16530
  2774 ——> 121050 TSR # 1, @ 16532
  2776 ——> 1
  3000 ——> 16532
043002 ——> 000010 RTRN : RTS PC
```

Patches on KDM

```
*015224 ——> 100
     26 ⎫
     30 ⎪
     32 ⎪
     34 ⎪
     36 ⎬ 100          Noop
     40 ⎪
     42 ⎪
     44 ⎭
```

APPENDIX 'B'

```
   PROGRAM  DBMS   DMDL ANALYSER FOR TDC316 IMPLETATION              DMD00010
   PROGRAM  DBMS   DMDL ANALYSER FOR TDC316 IMPLETATION              DMD00020
   THIS PROGRAM SCANS INPUT FROM DEVICE 5 FOR DMDL CARDS             DMD00040
************************************************************************DMD00030
************************************************************************DMD00050
   EXPLANATION OF PARAMETERS USED IN DMDL ANALYSER ARE--------       DMD00060
   RELID   -----RELATION IDENTIFICATION NUMBER                       DMD00070
   FLDID   -----FIELD/ITEM IDENTIFICATION NUMBER                     DMD00080
   AN      -----USER CODE FOR ALPHANUMERIC FORMAT SINGLE 20 CHARACTERDMD00090
   AD      -----USER CODE FOR ALPHANUMERIC FORMAT DOUBLE 60 CHARACTERDMD00100
   IN      -----USER CODE FOR INTEGER FORMAT  I5 MAX PERMISSIBLE     DMD00110
   RFDIR   -----RELATION - FIELDS/ITEMS DIRECTORY HAS ALL DETAILS    DMD00120
   RTAB    -----RELATION TABLES CONTAINS DETAILS OF EACH RELATION    DMD00130
   FTAB    -----FIELD TABLE CONTAINS DETAILS OF ALL FIELDS/ITEMS     DMD00140
   RELN    -----DMDLANGUAGE  CARD GIVING RELATION NAME ACCES CODE    DMD00150
   FLDS    -----DMDL FIELD DEFINITION CARD GIVING NAME TYPE          DMD00160
   DATA    -----DMDL DATA DEFINITION CARD GIVING NUMBERS DATA CARDS  DMD00170
   IEND    -----DMDL END CARD INDICATING PHYSICAL END OF ALL DATA    DMD00180
   CONT    -----CONTROL VARIABLE FOR IOPT ROUTINE  1--READ  2-WRITE  DMD00190
   DNO     -----DEVICE NUMBER  1--CR/LP  2--13  3--14  4--15  5--16  DMD00200
   DTNO    -----TEMPORARY DEVICE NUMBER                              DMD00210
   BUFF    -----SYSTEM BUFFER AREA USED BY IOPT ROUTINE              DMD00220
   ACCD    -----ACCES CODE   1--ALPHA A2  2-- LEVEL IN I2 FORMAT     DMD00230
   NI      -----NUMBER OFFIELDS HAVING   INTEGER  I5 FORMAT          DMD00240
   NA      -----NUMBER OFFIELDS HAVING   SINGLE ALPHANUMERIC 10A2    DMD00250
   DA      -----NUMBER OFFIELDS HAVING   DOUBLE ALPHANUMERIC 30A2    DMD00260
   KARD    -----INPUT FROPM  CARD READER/ KEYBOARD IN  A2 FORMAT     DMD00270
   DBA     -----INTEGER CODE FROM  DBA DATA BASE ADMINISTATOR        DMD00280
   IREL    ----RELATION NAMES REQUIRED BY USER STORED 2A2 FORMAT     DMD00290
   IRID    -----RELATION ID REUQIRED BY USER IN  I2 FORMAT           DMD00300
   IFID    -----FIELD ID REQUIRED BY USER STORED IN I2 FORMAT        DMD00310
   MCFR    -----COMMON RELATION -FIELD/ITEM MATRIX COTIANING FLDID   DMD00320
   NR      -----POINTER INDICATING CURRENT VALUE IN ----RTAB         DMD00330
   NF      -----POINTER INDICATING CURRENT VALUE IN ----FTAB         DMD00340
   NFR     -----POINTER INDICATING CURRENT VALUE IN ----RFDIR        DMD00350
   L       -----LOCATION ON DISK RELATIVE TO FIRST ENTRYON A DNO     DMD00360
   NOR     -----NUMBER OR POINTER TO A PARTICULAR TUPLE IN THE RELATNDMD00370
   NDC     -----NUMBER OF DATA CARDS / TUPLES/OCCURANCES IN A RELATINDMD00380
   NRQ     -----2*NO TWICE THE NUMBER OF RELATIONS REQUIRED BY USER  DMD00390
   IFLG A FLAG FOR DISCRIMINATING READ ONLY USERS 1--READ ONLY 2-ELSEDMD00400
   INTEGER ,DIMENSION AND COMMON CARDS FOR EACH OF THE VARIABLES     DMD00410
************************************************************************DMD00420
   INTEGER RELID,FLDID,AN,RFDIR,RTAB,FTAB,DATA,IEND,RELN,FLDS,TLRCW,
  1CONT,DNO,DTNO,BUFF,ACCD,AD,DA,DBA
   DIMENSION KARD(40),FTAB( 40,4),RFDIR(50,3),RTAB(20,10),BUFF( 2,40)
  1, IREL(20),IRID(10),ACCD(2),MCFR(10,10),IFID(10),DBA(2)
   COMMON KARD,NR,NF,NI,NA,FTAB,FLDID,RTAB,RELID,AN,IN,L,NDC,TLRCW,
  1CONT,DNO,NOR,NFR,BUFF,NRQ,IREL,IRID,IFLG,MCFR,RFDIR,ACCD,DA
  2 ,NAT,NII,NN,IFID
C***********************************************************************
C************    INITIALISATION OF VARIABLES AND CONSTANT    ************
C***********************************************************************
   REWIND  13
   REWIND  14
   REWIND  15
   REWIND  16
```

APPENDIX 'B'

```
      PROGRAM  DBMS   DMDL ANALYSER FOR TDC316 IMPLETATION              DMD00010
C     PROGRAM  DBMS   DMDL ANALYSER FOR TDC316 IMPLETATION              DMD00020
C     THIS PROGRAM SCANS INPUT FROM DEVICE 5 FOR DMDL CARDS             DMD00040
C***********************************************************************DMD00030
C***********************************************************************DMD00050
C     EXPLANATION OF PARAMETERS USED IN DMDL ANALYSER ARE--------       DMD00060
C     RELID    -----RELATION IDENTIFICATION NUMBER                      DMD00070
C     FLDID    -----FIELD/ITEM IDENTIFICATION NUMBER                    DMD00080
C     AN       -----USER CODE FOR ALPHANUMERIC FORMAT SINGLE 20 CHARACTERDMD00090
C     AD       -----USER CODE FOR ALPHANUMERIC FORMAT DOUBLE 60 CHARACTERDMD00100
C     IN       -----USER CODE FOR INTEGER FORMAT  I5 MAX PERMISSIBLE    DMD00110
C     RFDIR    -----RELATION - FIELDS/ITEMS DIRECTORY HAS ALL DETAILS   DMD00120
C     RTAB     -----RELATION TABLES CONTAINS DETAILS OF EACH RELATION   DMD00130
C     FTAB     -----FIELD TABLE CONTAINS DETAILS OF ALL FIELDS/ITEMS    DMD00140
C     RELN     -----DMDLANGUAGE  CARD GIVING RELATION NAME ACCES CODE   DMD00150
C     FLDS     -----DMDL FIELD DEFINITION CARD GIVING NAME TYPE         DMD00160
C     DATA     -----DMDL DATA DEFINITION CARD GIVING NUMBERS DATA CARDS DMD00170
C     IEND     -----DMDL END CARD INDICATING PHYSICAL END OF ALL DATA   DMD00180
C     CONT     -----CONTROL VARIABLE FOR IOPT ROUTINE  1--READ  2-WRITE DMD00190
C     DNO      -----DEVICE NUMBER  1--CR/LP  2--13  3--14  4--15  5--16 DMD00200
C     DTNO     -----TEMPORARY DEVICE NUMBER                             DMD00210
C     BUFF     -----SYSTEM BUFFER AREA USED BY IOPT ROUTINE             DMD00220
C     ACCD     -----ACCES CODE   1--ALPHA A2  2-- LEVEL IN I2 FORMAT    DMD00230
C     NI       -----NUMBER OFFIELDS HAVING   INTEGER  I5 FORMAT         DMD00240
C     NA       -----NUMBER OFFIELDS HAVING   SINGLE ALPHANUMERIC 10A2   DMD00250
C     DA       -----NUMBER OFFIELDS HAVING   DOUBLE ALPHANUMERIC 30A2   DMD00260
C     KARD     -----INPUT FROPM  CARD READER/ KEYBOARD IN  A2 FORMAT    DMD00270
C     DBA      -----INTEGER CODE FROM  DBA DATA BASE ADMINISTATOR       DMD00280
C     IREL     ----RELATION NAMES REQUIRED BY USER STORED 2A2 FORMAT    DMD00290
C     IRID     -----RELATION ID REUQIRED BY USER IN  I2 FORMAT          DMD00300
C     IFID     -----FIELD ID REQUIRED BY USER STORED IN I2 FORMAT       DMD00310
C     MCFR     -----COMMON RELATION -FIELD/ITEM MATRIX COTIANING FLDID  DMD00320
C     NR       -----POINTER INDICATING CURRENT VALUE IN ----RTAB        DMD00330
C     NF       -----POINTER INDICATING CURRENT VALUE IN ----FTAB        DMD00340
C     NFR      -----POINTER INDICATING CURRENT VALUE IN ----RFDIR       DMD00350
C     L        -----LOCATION ON DISK RELATIVE TO FIRST ENTRYON A DNO    DMD00360
C     NOR      -----NUMBER OR POINTER TO A PARTICULAR TUPLE IN THE RELATNDMD00370
C     NDC      -----NUMBER OF DATA CARDS / TUPLES/OCCURANCES IN A RELATINDMD00380
C     NRQ      -----2*NQ TWICE THE NUMBER OF RELATIONS REQUIRED BY USER DMD00390
C     IFLG A FLAG FOR DISCRIMINATING READ ONLY USERS 1--READ ONLY 2-ELSEDMD00400
C     INTEGER ,DIMENSION AND COMMON CARDS FOR EACH OF THE VARIABLES     DMD00410
C***********************************************************************DMD00420
      INTEGER RELID,FLDID,AN,RFDIR,RTAB,FTAB,DATA,IEND,RELN,FLDS,TLRCW,
     1CONT,DNO,DTNO,BUFF,ACCD,AD,DA,DBA
      DIMENSION KARD(40),FTAB( 40,4),RFDIR(50,3),RTAB(20,10),BUFF( 2,40)
     1, IREL(20),IRID(10),ACCD(2),MCFR(10,10),IFID(10),DBA(2)
      COMMON KARD,NR,NF,NI,NA,FTAB,FLDID,RTAB,RELID,AN,IN,L,NDC,TLRCW,
     1CONT,DNO,NOR,NFR,BUFF,NRQ,IREL,IRID,IFLG,MCFR,RFDIR,ACCD,DA
     2 ,NAT,NII,NN,IFID
C***********************************************************************
C*************   INITIALISATION OF VARIABLES AND CONSTANT   ************
C***********************************************************************
      REWIND  13
      REWIND  14
      REWIND  15
      REWIND  16
      IN=20041
      AN=20033
      IBL2=8224
      FLDS=19526
      RELN=17746
      DATA=16708
      IEND=17737
      NR=0
      NF=0
      NFR=0
      RELID=0
      FLDID=0
      L=1
      NDC=0
      DNO=1
      NOR=1
       NAT=0
      NN=0
       NII=0
      DBA(1)=16964
      DBA(2)=01
C     PROGRAM TO TEST DBA AUTHORISATION
      WRITE(2,2001)
 2001 FORMAT(10X,*MAY I HAVE YOUR IDENTIFICATION  PLEASE ?*)
      READ(1,2002) ACCD(1),ACCD(2)
      WRITE(2,2002)ACCD(1),ACCD(2)
 2002 FORMAT(A2,I2)
      IF ((ACCD(1).EQ.DBA(1)).AND.(ACCD(2).EQ.DBA(2)))GO TO 7
      WRITE(2,2003)
 2003 FORMAT(10X,* SORRY ONLY DBA IS ENTITLED TO USE DMD ANALYSER *)
      STOP
  7   WRITE(2,2009) ACCD(1),ACCD(2)
 2009 FORMAT(10X,*WELCOME SIR !DBA AUTHORISED CODE--- *,A2,I2)
      WRITE(2,2000)
 2000 FORMAT(10X,*BUILDING A RELATIONAL DATA BASE FOR DRDO LABS*,/,10X,
     145(1H-) )
C*****************************************************************************
X     WRITE(6,1000)
X1000 FORMAT(1H1,40X,*B U I L D I N G   R E L A T I O N A L   D A T A
     1B A S E *)
C    BLANKING    ALL    THE    TABLES      ***********************************
      DO 11 I=1,100
      DO 11 J=1,4
  11  FTAB(I,J)=IBL2
      DO 12 I=1,50
      DO 12 J=1,3
  12  RFDIR(I,J)=IBL2
      DO 13 I=1,20
      DO 13 J=1,10
  13  RTAB(I,J)=IBL2
C*******  THE MAIN PROGRAM FOR DDL SCANNER COMMENCES FROM HERE  *************
C
      WRITE(2,2004)
 2004 FORMAT(10X,* PLEASE GIVE RELATION DEFINITION AS RELN  NAME  AC?*)
  51  READ(1,1)(KARD(I),I=1,7)
   1  FORMAT(7A2)
```

```
      WRITE(2,2004)
 2004 FORMAT(10X,* PLEASE GIVE RELATION DEFINITION AS RELN  NAME  AC?*)
  51  READ(1,1)(KARD(I),I=1,7)
   1  FORMAT(7A2)
      WRITE(2,6) (KARD(I),I=1,7)
X     WRITE(6,1)(KARD(I),I=1,7 )
   6  FORMAT(7A2)
      IF(KARD(1).EQ.RELN) CALL IRELN
       IF(KARD(1).NE.RELN) GO TO 51
C****HAVING RECOGNISED DDL CARD CALL IRELN ALLOCATE DEVICE NO  CHECK FLDS*******
      WRITE(2,2005)
 2005 FORMAT(10X,*FIELD DEFINITION AS  FLDS  NAME  IN OR AN OR AD *,/
     1  10X,*AT END GIVE NO OF DATA CARDS AS  DATA  XX 4 BLANKS*)
   2  READ(1,1) (KARD(I),I=1,7)
      WRITE(2,6) (KARD(I),I=1,7)
X     WRITE(6,1)(KARD(I),I=1,7 )
   3  IF(KARD(1).NE.FLDS) GO TO 4
      CALL IFLDS
      GO TO 2
   4  IF(KARD(1).EQ.DATA) GO TO 5
      GO TO 2
C********  RELATION TABLE FIELD TABLE COMPLETED GOING TO READ DATA  ************
   5  NDC=KARD(4)
      CALL ICON(NDC)
      TLRCW=5*NI+20*NA+60*DA
X     WRITE(6,1101)    TLRCW
X1101 FORMAT(50X,*TOTAL LENGTH OF EACH RECORD ----*,I5)
C
      WRITE(2,2006) NDC
 2006 FORMAT(10X,*PLACE*,I5, *   DATA CARDS IN CARD READER!*)
C********  ALLOCATING  AREA FROM STARTING ADDRESS L TLRCW FOR EACH REC  ********
      RTAB(NR,5)=L
      RTAB(NR,7)=NDC
      RTAB(NR,8)=NI
      RTAB(NR,9)=NA
      RTAB(NR,10)=DA
X     WRITE(6,1002)(RTAB(NR,J),J=1,10),NR
X1002 FORMAT(10X,*RELID NAME DNO STPN  AC NDC  NI  NA  DA*,/,
     1  10X,I3,3X,2A2,2I4,3X,A2,2X,I2,2X,I2,2X,I2,2X,I2,* RTAB---*,I2)
C*******  TESTING FOR IEND  CARD  AFTER READING THE DATA CARDS  ***************
      DTNO=DNO
      NOR=1
      I=1
  10  CONT=1
      DNO=1
      CALL IOPT
      CONT=2
      DNO=DTNO
      CALL IOPT
      L=L+1
      I=I+1
      IF(  I.LE.NDC) GO TO 10
C         INCORPORATING FOR OVERFLOW AREA FOR ADDITOIN OF 10 % RECORDS
      IOFL=NDC/10+1
      L=L+IOFL
C           ADJUSTING BUFFER TO HAVE SAMILIAR FIELDS
      DO 15 J=1,NN
      IF(J.LE. NI) BUFF(1,J)=0
      IF(J.GT.NI) BUFF(1,J)=IBL2
   15 CONTINUE
C      WRITING ON THE SAME DEVICE NUMBER
      CONT=2
      DNO=DTNO
      DO 17 K=1,IOFL
   17 CALL IOPT
C********  HAVING WRITTEN ON DISK  A FULL RELATION TESTING FOR IEND  **********
      WRITE(2,2007)
 2007 FORMAT(10X,* IS IT ALL? GIVE IEND CARD ELSE GIVE RELN  CARD !*)
    9 READ(1,1)(KARD(I),I=1,7)
      WRITE(2,6) (KARD(I),I=1,7)
X     WRITE(6,1)(KARD(I),I=1,7 )
      IF(KARD(1).EQ.IEND) GO TO 20
      IF(KARD(1).NE.RELN) GO TO 30
      CALL IRELN
       GO TO 2
C ********  ALL RELATIONS HAVING BEEN DEFINED BOOK KEEPING CAN BE ADDED *******
   20 CONTINUE
X     WRITE(6,25)
X 25  FORMAT(1H1,40X,* A L L   R E L A T I O N   I N   D A T A B A S E *)
   21 WRITE(2,2008)
 2008 FORMAT(20X,* ALL RELATION IN DATA BASE *,/,20X,25(1H-))
 1006 FORMAT(///25X,*CONTENTS  OF  RTAB *)
 1009 FORMAT(10X,I5,2A2,2I5,A2,4I5)
 1111 FORMAT(20X,*NUMBER OF ENTRIES IN TABLE--*,I5)
X     WRITE( 6, 1006)
X     WRITE(6,1009) ((RTAB(I,J),J=1,10),I=1,NR)
X     WRITE(6,1111)      NR
X     WRITE( 6, 1007)
 1007 FORMAT(//,25X,*CONTENTS  OF  FTAB*,/20X,*FLDID      NAME   FORMAT*)
X      WRITE(6,1010) ((FTAB(IF,JF),JF=1, 4),IF=1,NF)
 1010 FORMAT(20X,I5,5X,2A2,5X,I5)
X     WRITE(6,1111)     NF
X     WRITE( 6, 1008)
 1008 FORMAT(//25X,*CONTENTS  OF  RFDIR*,/10X,*RELID          FLDID
    1     FORMAT*)
X     WRITE(6,1011) ((RFDIR(IR,JR),JR=1,  3),IR=1,NFR)
 1011 FORMAT(10X,I5,10X,I5,10X,I5)
X     WRITE(6,1111)     NFR
      WRITE(2,1012)
X     WRITE(6,1012)
 1012 FORMAT(10X,60(1H-),/,20X,*DMD ANALYSER COMPLETED*,/,10X,60(1H-))
C*******      WRITING  ALL  TABLES    ON    DISK  UNIT    16      ***************
      REWIND  16
      WRITE(16,501)NR,NF,NFR
  501 FORMAT(3I2)
      WRITE(16,9009)((RTAB(I,J),J=1,10),I=1,NR)
 9009 FORMAT(I5,2A2,2I5,A2,4I5)
      WRITE(16,9010) ((FTAB(IF,JF),JF=1, 4),IF=1,NF)
 9010 FORMAT(I5,2A2,I5)
      WRITE(16,9011)((RFDIR(IR,JR),JR=1,  3),IR=1,NFR)
       REWIND  16
 9011 FORMAT(3I5)
       STOP
  30  WRITE(2,40)
  40  FORMAT(10X,*------ERROR---- RELATION NOT PROPERLY DEFINED------- *)
      GO TO 9
 100  FORMAT(NI5,NA2)
       STOP
      END
C
C
C
      SUBROUTINE   IRELN
C
C**********************************************************************************
C*******  THIS SUBROUTINE BUILDS RTAB BY ALLOCATING RELID AND DNO    ***********
C**********************************************************************************
      INTEGER RELID,FLDID,AN,RFDIR,RTAB,FTAB,DATA,IEND,RELN,FLDS,TLRCW,
     1CONT,DNO,DTNO,BUFF,ACCD,AD,DA,DBA
      DIMENSION KARD(40),FTAB(100,4),RFDIR(50,3),RTAB(20,10),BUFF(50,40)
     1, IREL(20),IRID(10),ACCD(2),MCFR(10,10),IFID(10),DBA(2)
      COMMON KARD,NR,NF,NI,NA,FTAB,FLDID,RTAB,RELID,AN,IN,L,NDC,TLRCW,
     1CONT,DNO,NOR,NFR,BUFF,NRG,IREL,IRID,IFLG,MCFR,RFDIR,ACCD,DA
     2 ,NAT,NII,NN,IFID
C**********************************************************************************
C*******     TESTING FOR  DUPLICITTY IN RELATION  NAMES           ***************
C**********************************************************************************
      NI=0
      NA=0
      DA=0
      DO 10 I=1,NR
      IF(KARD(4).EQ.RTAB(I,2).AND.(KARD(5).EQ.RTAB(I,3))) GO TO 30
 10   CONTINUE
C*******  ALLOCATING  DEVICE NO  ON WHICH TE RELATION IS TO BE STORED *********
      NR =NR+1
      IF(NR.EQ . 7) L=1
      IF(NR.EQ .13) L=1
      IF(NR.EQ .19) L=1
      RELID=RELID+1
      RTAB(NR,1) =RELID
      RTAB(NR,2)=KARD(4)
      RTAB(NR,3)=KARD(5)
      IF(NR.LE.6) DNO=4
      IF((NR.GT.6).AND.(NR.LE.12))DNO=3
        IF (NR.GT.12) DNO=2
      RTAB(NR,4)=DNO
      RTAB(NR,6)=KARD(7)
      RETURN
 30   WRITE(2,35)
 35   FORMAT(10X,*------ERROR----- DUPLICATE RELATION NAME CONTACT DBA *)
```

```
      SUBROUTINE ICON(IGIV)
C
C***********************************************************************
C********  THIS SUBROUTINE  CCONVERTS   A   FORMAT   INTO  I5 FORMAT*********
C***********************************************************************
       I=IGIV
      J=I
      I=I/256-48
      J=J-(I+48)*256
      I=(J-48)*10+I
      IGIV=I
      RETURN
      END
      SUBROUTINE    IOPT
C***********************************************************************
C*****THIS SUBROUTINE READS / WRITES ON ADEVICE IN VARIABLE FORMAT *********
C********  CONT=1 FOR READ 2 FOR WRITE DNO=1 FORCR/LP.2.3.4= 5 FOR DISK *******
C***********************************************************************
      INTEGER RELID,FLDID,AN,RFDIR,RTAB,FTAB,DATA,IEND,RELN,FLDS,TLRCW,
     1CONT,DNO,DTNO,BUFF,ACCD,AD,DA,DBA
      DIMENSION KARD(40),FTAB(100,4),RFDIR(50,3),RTAB(20,10),BUFF(50,40)
     1, IREL(20),IRID(10),ACCD(2),MCFR(10,10),IFID(10),DBA(2)
      COMMON KARD,NR,NF,NI,NA,FTAB,FLDID,RTAB,RELID,AN,IN,L,NDC,TLRCW,
     1CONT,DNO,NOR,NFR,BUFF,NRQ,IREL,IRID,IFLG,MCFR,RFDIR,ACCD,DA
     2 ,NAT,NII,NN,IFID
      I=NOR
      NAT=10*NA+30*DA
      NN=NI+NAT
      NII=NI+1
      IF(CONT.EQ.2) GO TO 20
      GO TO (6,13,14,15,16),DNO
 6    READ ( 5,100)NI,(BUFF(I,J),J=1,NI),NAT,(BUFF(I,J),J=NII,NN)
      RETURN
 13   FIND13,L
      READ (13,100)NI,(BUFF(I,J),J=1,NI),NAT,(BUFF(I,J),J=NII,NN)
      ENDFILE 13
      RETURN
 14   FIND14,L
      READ (14,100)NI,(BUFF(I,J),J=1,NI),NAT,(BUFF(I,J),J=NII,NN)
       ENDFILE 14
      RETURN
 15   FIND15,L
      READ (15,100)NI,(BUFF(I,J),J=1,NI),NAT,(BUFF(I,J),J=NII,NN)
      ENDFILE 15
      RETURN
 16   FIND16,L
      READ (16,100)NI,(BUFF(I,J),J=1,NI),NAT,(BUFF(I,J),J=NII,NN)
      ENDFILE 16
      RETURN
 20   IF((IFLG.EQ.1).AND.(DNO.GT.1)) GO TO 420
      GO TO (108,113,114,115,116),DNO
 108  WRITE( 6,100)NI,(BUFF(I,J),J=1,NI),NAT,(BUFF(I,J),J=NII,NN)
      RETURN
 113  FIND13,L
```

```
      WRITE(13,100)NI,(BUFF(I,J),J=1,NI),NAT,(BUFF(I,J),J=NII,NN)
X     WRITE( 6,100)NI,(BUFF(I,J),J=1,NI),NAT,(BUFF(I,J),J=NII,NN)
      ENDFILE 13
      RETURN
 114  FIND14,L
      WRITE(14,100)NI,(BUFF(I,J),J=1,NI),NAT,(BUFF(I,J),J=NII,NN)
X     WRITE( 6,100)NI,(BUFF(I,J),J=1,NI),NAT,(BUFF(I,J),J=NII,NN)
      ENDFILE 14
      RETURN
 115  FIND15,L
      WRITE(15,100)NI,(BUFF(I,J),J=1,NI),NAT,(BUFF(I,J),J=NII,NN)
X     WRITE( 6,100)NI,(BUFF(I,J),J=1,NI),NAT,(BUFF(I,J),J=NII,NN)
      ENDFILE 15
      RETURN
 116  FIND16,L
      WRITE(16,100)NI,(BUFF(I,J),J=1,NI),NAT,(BUFF(I,J),J=NII,NN)
      ENDFILE 16
      RETURN
 420  CONTINUE
      WRITE(2,421)
X     WRITE(6,421)
 421  FORMAT(10X,* SORRY  USER   YOU ARE NOT AUTHORISED TO WRITE IN DB*)
 100  FORMAT(NI5,NA2)
       RETURN
      END
C
C
C

      SUBROUTINE  IFLDS
C
C*************************************************************************************
C********  THIS SUBROUTINE ALLOCATES DISTINCT  FDID FOR EACH FD   **************
C*************************************************************************************
      INTEGER RELID,FLDID,AN,RFDIR,RTAB,FTAB,DATA,IEND,RELN,FLDS,TLRCW,
     1CONT,DNO,DTNO,BUFF,ACCD,AD,DA,DBA
      DIMENSION KARD(40),FTAB(100,4),RFDIR(50,3),RTAB(20,10),BUFF(50,40)
     1, IREL(20),IRID(10),ACCD(2),MCFR(10,10),IFID(10),DBA(2)
      COMMON KARD,NR,NF,NI,NA,FTAB,FLDID,RTAB,RELID,AN,IN,L,NDC,TLRCW,
     1CONT,DNO,NOR,NFR,BUFF,NRQ,IREL,IRID,IFLG,MCFR,RFDIR,ACCD,DA
     2 ,NAT,NII,NN,IFID
      AD=17473
      NF=NF+1
      NFR=NFR+1
      DO 10 I=1,NF
       IF((KARD(4).EQ.FTAB(I,2)).AND.(KARD(5).EQ.FTAB(I,3)))GO TO 30
 10   CONTINUE
       FLDID=FLDID+1
C********   STORING IN FTAB FLDID AND NAME OF THE   FIELD
      FTAB(NF,1)=FLDID
      FTAB(NF,2)=KARD(4)
      FTAB(NF,3)=KARD(5)
C ****** TESTING FOR TYPE OF FORMAT AND STORING1 FOR 2 FOR AN  ************
C************KEEP I5   FOR INTEGER FORMAT AND A2 FOR ALPHANUMERIC FID***********
```

```
        IF(KARD(7).EQ.IN) GO TO 20
       IF( KARD(7).EQ. AN) GO TO 25
       IF(KARD(7).EQ.AD) GO TO 27
       GO TO 50
 20     IFOT=1
       FTAB(NF,4)=IFOT
       NI=NI+1
       GO TO 40
 25    IFOT=2
       FTAB(NF,4)=IFOT
         NA=NA+1
       GO TO 40
 27    IFOT=3
       FTAB(NF,4)=IFOT
       DA=DA+1
       GO TO 40
C********  ENTERING INRFDIR THE OCCURANCE OF RELATION FIELDS     ***************
 30    NF=NF-1
       IFDID=FLDID
       FLDID=FTAB(I,1)
       IFOT=FTAB(I,4)
       IF(KARD(7).EQ.IN) GO TO 45
       IF(KARD(7).EQ.AN) GO TO 55
       IF(KARD(7).EQ.AD) GO TO 56
       GO TO 50
 45    IFOT=1
       NI=NI+1
       GO TO 57
 55    IFOT=2
       NA=NA+1
       GO TO 57
 56    IFOT=3
       DA=DA+1
 57    RFDIR(NFR,1)=RELID
       RFDIR(NFR,2)=FLDID
       RFDIR(NFR,3)=IFOT
       FLDID=IFDID
       RETURN
 40    RFDIR(NFR,1)=RELID
       RFDIR(NFR,2)=FLDID
       RFDIR(NFR,3)=IFOT
       RETURN
 50    CONTINUE
X      WRITE(6,60)
       WRITE(2,60)
 60    FORMAT( 10X,*----------ERROR-----TYPE OF FD NOT INDICATE ON FLDS*)
       NF=NF-1
       NFR=NFR-1
       FLDID=FLDID-1
       RETURN
       END
```

```
      PROGRAM    DBMS    DML  SEARCH  ROUTINES
C**************************************************************************
C     THIS IS THE MAIN PROGRAM FOR TESTING ALL ROUTNESOF DSLAND DML
C**************************************************************************
C     EXPLANATION OF VARIBLES USED FOLLOWS---------------
C     RELID   -----RELATION IDENTIFICATION NUMBER
C     FLDID   -----FIELD/ITEM IDENTIFICATION NUMBER                     DMD00070
C     AN      -----USER CODE FOR ALPHANUMERIC FORMAT SINGLE 20 CHARACTERDMD00080
C     AD      -----USER CODE FOR ALPHANUMERIC FORMAT DOUBLE 60 CHARACTERDMD00090
C     IN      -----USER CODE FOR INTEGER FORMAT  I5 MAX PERMISSIBLE     DMD00 00
C     RFDIR   -----RELATION - FIELDS/ITEMS DIRECTORY HAS ALL DETAILS    DMD00  0
C     RTAB    -----RELATION TABLES CONTAINS DETAILS OF EACH RELATION    DMD00120
C     FTAB    -----FIELD TABLE CONTAINS DETAILS OF ALL FIELDS/ITEMS     DMD001 0
C     CONT    -----CONTROL VARIABLE FOR IOPT ROUTINE  1--READ  2-WRITE  DMD00140
C     DNO     -----DEVICE NUMBER  1--CR/LP  2--13  3--14  4--15  5--16  DMD00190
C     BUFF    -----SYSTEM BUFFER AREA USED BY IOPT ROUTINE              DMD00200
C     ACCD    -----ACCES CODE   1--ALPHA A2  2-- LEVEL IN I2 FORMAT     DMD00220
C     NI      -----NUMBER OFFIELDS HAVING   INTEGER  I5 FORMAT          DMD002 0
C     NA      -----NUMBER OFFIELDS HAVING   SINGLE ALPHANUMERIC 10A2    DMD00240
C     DA      -----NUMBER OFFIELDS HAVING   DOUBLE ALPHANUMERIC 30A2    DMD00250
C     KARD    -----INPUT FROPM  CARD READER/ KEYBOARD IN  A2 FORMAT     DMD00260
C     DBA     -----INTEGER CODE FROM  DBA DATA BASE ADMINISTATOR        DMD00270
C     IREL    ----RELATION NAMES REQUIRED BY USER STORED 2A2 FORMAT     DMD00280
C     IRID    -----RELATION ID REUQIRED BY USER IN  I2 FO-MAT           DMD00290
C     IFID    -----FIELD ID REQUIRED BY USER STORED IN I2 FORMAT        DMD00300
C     MCFR    -----COMMON RELATION -FIELD/ITEM MATRIX COTIANING FLDID   DMD00310
C     NR      -----POINTER INDICATING CURRENT VALUE IN ----RTAB         DMD00320
C     NF      -----POINTER INDICATING CURRENT VALUE IN ----FTAB         DMD00
C     NFR     -----POINTER INDICATING CURRENT VALUE IN ----RFDIR        DMD00340
C     L       -----LOCATION ON DISK RELATIVE TO FIRST ENTRYON A DNO     DMD00 50
C     NOR     -----NUMBER OR POINTER TO A PARTICULAR TUPLE IN THE RELATNDMD00 60
C     NDC     -----NUMBER OF DATA CARDS / TUPLES/OCCURANCES IN A RELATINDMD00 70
C     NRQ     -----2*NQ TWICE THE NUMBER OF RELATIONS REQUIRED BY USER  DMD00  0
C     IFLG A FLAG FOR DISCRIMINATING READ ONLY USERS 1--READ ONLY 2-ELSEDMD003 0
C     IUSR    -----ARRAY OF USER AREA FOR DATA MANIPULATION             DMD00400
C**************************************************************************
      INTEGER RELID,FLDID,RFDIR,RTAB,FTAB,DNO,BUFF,ACCD,DA,CONT,DBA
      DIMENSION KARD(40),FTAB( 40,4),RFDIR(50,3),RTAB(20,10),BUFF( 2,40)
     1,IREL(20),IRID(10),ACCD(2),MCFR(10,10),IFID(10),IUSR(40,40),DBA(2)
      DIMENSION IV(30),IC(30),IG(30),ITRD(10),ITFD(10)
      DIMENSION IABC(2),ITV(40),MNAM(2),MFNAM(20)
      COMMON KARD,NR,NF,NI,NA,FTAB,FLDID,RTAB,RELID,L,NDC,CONT,DNO,NOR
     1 ,NFR,BUFF,NRQ,IREL,IRID,IFLG,MCFR,RFDIR,ACCD, DA,NAT,NII,NN,IFID
     2 ,IUSR,ITFLG,DBA
C********   INIATIALISATION OF DBMS PARAMETERS      FOR  DML   **************
      IABC(1)=16961
      IABC(2)=01
      ACCD(1)=16961
      ACCD(2)=01
      IF LG=1
      REWIND  13
      REWIND  14
      REWIND  15
      REWIND  16
      WRITE(2,1)
X     WRITE(6,1)
   1  FORMAT(     10X,*D  S  L1 T E S T  P R O G R A M S  O U T P U T*)
C*************************************************************************
C*****   TESTING  SUBROUTINE  OPEN  ******************************************
      CALL   OPEN                                      ************************
      NQ=NRQ/2
X     WRITE(6,1103)
 1103 FORMAT(20X,*CONTENTS   OF    MCFR  *,/ )
X     WRITE(6,1104) NQ, ((MCFR(I,J), J=1,NQ), I=1,NQ)
 1104 FORMAT(10X,NI5)
X     WRITE(6,1112)
      WRITE(2,1112)
 1112 FORMAT(10X,50(1H-),/,20X,*  OPEN  TEST COMPLETED*,/,10X,50(1H-))
      PAUSE   1
C**********     TESTING    SUBROUTINE      BSERCH    **************************
       K=2
         I=11
       WRITE(2,1203) K
 1203  FORMAT(10X,*GIVE LOCATION OF RELID--*,I3,*  IN RTAB ?*)
       CALL BSERCH (K,RTAB,I,INDX)
       WRITE(2,1202) K,INDX
 1202  FORMAT(10X,*RELID *,I3,*  MATCHED IN RTAB AT --*,I3)
X      WRITE(6,1201)
       WRITE(2,1201)
 1201  FORMAT(10X,50(1H-),/,20X,* BSERCH TEST COMPLETED*,/,10X,50(1H-))
       PAUSE 2
C********* TESTING  SUBROUTINE   GETO       *************************
        ID=1
       IL=4
       IT=1
       WRITE(2,1401)IL,ID,IT
       CALL GETO(ID,IL,IT)
       N=1
       WRITE( 2,100)NI,(IUSR(N,M),M=1,NI),NAT,(IUSR(N,M),M=NII,NN)
       ID=3
       IL=6
       IT=0
       WRITE(2,1401)IL,ID,IT
       CALL GETO(ID,IL,IT)
       WRITE( 2,100)NI,(IUSR(N,M),M=1,NI),NAT,(IUSR(N,M),M=NII,NN)
       WRITE(2,1402)
X      WRITE(6,1402)
       REWIND 15
 1401  FORMAT(10X,*GET  REC OCCURENCE--*,I2,* OF RELID--*,I3,*REWND*,I2)
 1402  FORMAT(10X,50(1H-),/,20X,* GETO  TEST  COMPLETED*,/,10X,50(1H-))
       PAUSE   3
C*********   TESTING    SUBROUTINE      GETF         *********************
       IB=1
       IFZ=5
       IOR=6
       WRITE(2,1301) IFZ,IB,IOR
 1301  FORMAT(10X,*GET*,I5,*FLDID OF*,I2,*RELID*,I2,*REC OCCURRENCE ?*)
       CALL GETF(IB,IFZ,IOR,IPOSN,IV,IFO)
      WRITE(2,1306)IB,IFZ,IPOSN,IFO
 1306 FORMAT(5X,*GETF OP--RID-*,I5,*  FID*,I5,* FPOSN*,I5,* FOT*,I5)
       IF(IFO.GE.2)          GO TO 1312
      WRITE(2,1321) IUSR(1,1)
 1321  FORMAT(20X,I5)
      GO TO 222
 1312 IF(IFO.EQ.2)LNF=10
      IF(IFO.EQ.3)LNF=30
      WRITE     (2,1322) (IUSR(1,ML),ML=1,LNF )
 1322 FORMAT(5X,30A2)
 222  CONTINUE
X     WRITE(6,1302)
      WRITE(2,1302)
 1302 FORMAT(10X,50(1H-),/,20X,* GETF  TEST  COMPLETED*,/,10X,50(1H-))
      PAUSE    4
C********        TESTING   SUBROUTINE   G E T V     ****************************
      IRI=1
      IFI=5
      IRFI=6
      READ(5,1508)       (IG(I),I=1,30)
 1508 FORMAT(30A2)
      IP=5
      WRITE(2,1509)IRFI,IRI,IFI
 1509 FORMAT(10X,*GET FLDID--*,I3,*FROM RELID*,I2,*WHERE VALUE FID-*,I2)
      WRITE(2,1510) (IG(I),I=1,30)
 1510 FORMAT(10X,30A2)
      CALL GETV(IRI,IFI,IG,IRFI,IC,IFOR,IP)
      GO TO(1500,1501,1503),IFOR
 1500 CONTINUE
X     WRITE(6,1505) IC(1)
 1505 FORMAT(20X, I5)
      GO TO 1507
 1501 LNF=10
      GO TO 1504
 1503 LNF=30
 1504 CONTINUE
X     WRITE(6,1506) LNF,(IC(K),K=1,LNF)
 1506 FORMAT(20X,NA2)
 1507 CONTINUE
      WRITE(2,1506) LNF,((IUSR(N,M),M=1,LNF),N=2,6)
X     WRITE(6,1502)
      WRITE(2,1502)
 1502 FORMAT(10X,50(1H-),/,20X,* GETV   TEST COMPLETED*,/,10X,50(1H-))
      PAUSE    5
C********    TESTING  SUBRUOTINE   GETQ   WITH SINGLE QUAL  CONSTRAINT**********
      NRQ=6
      NQ=NRQ/2
      READ(5,2601)NRQ,(IRID(I),IFID(I),I=1,NQ          )
X     WRITE(6,2601)NRQ,(IRID(I),IFID(I),I=1,NQ)
 2601 FORMAT(NI2)
      IDQ=1
      IFQ=1
      IOPR=3
      IV(1)=1001
      ION=1
      WRITE(2,1701)IDQ,IFQ,IV(1)
      CALL GETQ(IDQ,IFQ,IOPR,IV,ION)
      WRITE(2,1702)
 1701 FORMAT(10X,*GIVEN RELID--*,I2,* FLDID--*,I2,* VALUE--*,I5)
X     WRITE(6,1702)
 1702 FORMAT(10X,50(1H-),/,20X,* GETQ  TEST  COMPLETED*,/,10X,50(1H-))
      PAUSE   6
C*********  TESTING  SUBROUTINE   GETR    ---  RETRIVING A REL  ***************
      IDQ=2
      WRITE(2,1801) IDQ
 1801 FORMAT(10X,*GET WHOLE RELATION WITH ID--*,I2)
      CALL GETR(IDQ,IABC)
X     WRITE(6,1802)
      WRITE(2,1802)
 1802 FORMAT(10X,50(1H-),/,20X,* GETR  TEST  COMPLETED*,/,10X,50(1H-))
C*********  TESTING   SUBROUTINE    GETU                  *******************
      NRQ=6
      NQ=NRQ/2
      READ(5,1601)NRQ,(ITRD(I),ITFD(I),I=1,NQ       )
X     WRITE(6,1601) NRQ,(ITRD(I),ITFD(I),I=1,NQ)
 1601 FORMAT(NI2)
      WRITE(2,105) NQ,(ITRD(J),J=1,NQ)
      WRITE(2,106) NQ,(ITFD(J),J=1,NQ)
  105 FORMAT(10X,*FROM RELATIONS ID--*,NI5)
  106 FORMAT(10X,*GETVALUES OF FLDIDS--*,NI5)
      CALL GETU  (ITRD,ITFD)
X     WRITE(6,1602)
      WRITE(2,1602)
 1602 FORMAT(10X,50(1H-),/,20X,* GETU  TEST COMPLETED*,/,10X,50(1H-))
      PAUSE   7
C*********  TESTING     SUBROUTINE     DUMP                ****************
      CALL DUMP(IABC)
X     WRITE(6,2002)
      WRITE(2,2002)
 2002 FORMAT(10X,50(1H-),/,20X,*CDUMP  TEST  COMPLETED*,/,10X,50(1H-))
C*********  TESTING     SUBROUTINE     ADDREC              ****************
      IRD=2
      WRITE(2,602) IRD
 602  FORMAT(10X,*RECORD TO ADDED IN RELID--*,I2,* FOLLOWS--*)
      READ (5,401)(ITV(J),J=1,5)
      WRITE(6,401)(ITV(J),J=1,5)
      WRITE(2,401)(ITV(J),J=1,5)
 401   FORMAT(6I5)
      CALL ADDREC(IRD,ITV,IABC)
      ION=11
      JK=1
      WRITE(2,603) ION,IRD
 603  FORMAT(10X,*TESTING IOR--*,I3,*IN RELID--*,I3)
      CALL GETO( IRD,ION,JK)
      WRITE(6,401) (IUSR(1,M),M=1,4)
      WRITE(2,401) (IUSR(1,M),M=1,4)
X     WRITE(6,2402)
      WRITE(2,2402)
 2402 FORMAT(20X,80(1H-),/,40X,*ADDREC TEST  COMPLETED*,/,20X,80(1H-))
C*********  TESTING     SUBROUTINE     MDYREC                  ****************
      MID=1
      MFD=5
      MOR=3
      WRITE(2,801) MID,MFD,MOR
  801 FORMAT(10X,*MODIFY RELID---*,I2,* FLDID---*,I3,*  IOR-*,I2,*BELOW-*)
      READ (5,301) (IC(J),J=1,30)
      WRITE(6,301) (IC(J),J=1,30)
      WRITE(2,301) (IC(J),J=1,30)
  301 FORMAT(40A2)
      CALL MDYREC(MID,MFD,MOR,IC,IABC)
      WRITE(2,802)
  802 FORMAT(10X,* TESTING THE NEW VALUE USING GETF*)
      CALL GETF(MID,MFD,MOR,IPN,IV,IFO)
      WRITE(2,301) (IV(K),K=1,20)
X     WRITE(6,2302)
      WRITE(2,2302)
 2302 FORMAT(20X,80(1H-),/,40X,*MDYREC TEST  COMPLETED*,/,20X,80(1H-))
C*********  TESTING     SUBROUTINE     DELTRL                  ****************
      IDD=5
      WRITE(2,901) IDD
  901 FORMAT(10X,*PROCEEDING TO DELET RELATION ID---*,I3)
      CALL DELTRL (IDD,IABC)
      WRITE( 6,9009)((RTAB(I,J),J=1,10),I=1,NR)
      WRITE( 6,9010) ((FTAB(IF,JF),JF=1, 4),IF=1,NF)
      WRITE( 6,9011)((RFDIR(IR,JR),JR=1,  3),IR=1,NFR)
X     WRITE(6,2202)
      WRITE(2,2202)
 2202 FORMAT(20X,80(1H-),/,40X,*DELTRL TEST  COMPLETED*,/,20X,80(1H-))
C*********  TESTING     SUBROUTINE     ADDREL                  ****************
      READ (5,701) MNAM(1),MNAM(2),MNDC,MNI,MNA,MDA,(MFNAM(I),I=1,12)
      WRITE(2,702)
  702 FORMAT(10X,*PARAMETERS OF THE NEW RELATION TO BE ADDED FOLLOW--*)
      WRITE(6,701) MNAM(1),MNAM(2),MNDC,MNI,MNA,MDA,(MFNAM(I),I=1,12)
      WRITE(2,701) MNAM(1),MNAM(2),MNDC,MNI,MNA,MDA,(MFNAM(I),I=1,12)
  701 FORMAT(2A2,4I2,12A2)
      READ (5,601) ((IUSR(I,J),J=1,33),I=2,11)
  601 FORMAT(3I5,30A2)
      CALL ADDREL(MNAM,MNDC,MFNAM,MNI,MNA,MDA,IABC)
      WRITE( 6,9009)((RTAB(I,J),J=1,10),I=1,NR)
      WRITE( 6,9010) ((FTAB(IF,JF),JF=1, 4),IF=1,NF)
      WRITE( 6,9011)((RFDIR(IR,JR),JR=1,  3),IR=1,NFR)
X     WRITE(6,2102)
      WRITE(2,2102)
 2102 FORMAT(20X,80(1H-),/,40X,*ADDREL TEST  COMPLETED*,/,20X,80(1H-))
C*********  TESTING     SUBROUTINE     CLOSE                   ****************
      CALL CLOSE(IABC)
X     WRITE(6,1902)
      WRITE(2,1902)
 1902 FORMAT(10X,50(1H-),/,20X,*CLOSE  TEST  COMPLETED*,/,10X,50(1H-))
  100 FORMAT(N15,NA2)
      STOP
      END
      SUBROUTINE OPEN
C**************************************************************************************
C*********   THIS SUBROUTINE CHECKS AUTHORASATION AND PREPARES MCFR ***********
CIREL STORES RELATION NAMES  NRQ  IS THE NUMBER OF RELATIONS REQUIRED
C********   ACCD  STORES THE ACESS CODE AND ACCES LEVEL                ***********
C*********  COMPARING  RELATION NAME  WITH NAME STORED IN  RTAB        ***********
C**************************************************************************************
      INTEGER RELID,FLDID,RFDIR,RTAB,FTAB,DNO,BUFF,ACCD,DA,CONT,DBA
      DIMENSION KARD(40),FTAB( 40,4),RFDIR(50,3),RTAB(20,10),BUFF( 2,40)
     1,IREL(20),IRID(10),ACCD(2),MCFR(10,10),IFID(10),IUSR(40,40),DBA(2)
      COMMON KARD,NR,NF,NI,NA,FTAB,FLDID,RTAB,RELID,L,NDC,CONT,DNO,NOR
     1 ,NFR,BUFF,NRQ,IREL,IRID,IFLG,MCFR,RFDIR,ACCD ,DA,NAT,NII,NN,IFID
     2 ,IUSR,ITFLG,DBA
X     WRITE(6,1001)
 1001 FORMAT(30X,*ENTERING  SUBROUTINE------OPEN  *)
C***********  INITIALISATION  OF  VARIABLES                     *********************
      IFLG=2
      IP=0
      DO 101 K=1,10
 101  IRID(K)=0
C     INCORPATING DIALOGUE WITH   TTY
      READ (16,501)NR,NF,NFR
      WRITE(6,501) NR,NF,NFR
  501 FORMAT(3I2)
 1002 FORMAT(3I2)
      WRITE(2,2002)
 2002 FORMAT(10X,*USER ! IDENTIFY  ACCES CODE IN A2 AND LEVEL IN I2 ?*)
      READ(1,1003) ACCD(1),ACCD(2)
      WRITE(2,1003) ACCD(1),ACCD(2)
 1003 FORMAT(A2,I2)
      WRITE(2,2003)
 2003 FORMAT(10X,*HOW MANY RELATIONS YOU WOULD BE USING ? *)
      READ(1,1004) NQ
      WRITE(2,1004) NQ
 1004 FORMAT(I2)
      NRQ=2*NQ
X     NQ=NRQ/2
      WRITE(2,2004)
 2004 FORMAT(10X,*RELATION NAME IN 2A2 FORMAT  PLEASE ? *)
      READ(1,1102) NRQ,(IREL(I),I=1,NRQ)
      WRITE(2,1102)NRQ,(IREL(I),I=1,NRQ)
 1102 FORMAT(NA2)
C
C*******  READING  RTAB ,FTAB, RFDIR  FOROM THE SYSTEM AREA    ON D4 *********
      READ (16,9009)((RTAB(I,J),J=1,10),I=1,NR)
 9009 FORMAT(I5,2A2,2I5,A2,4I5)
      READ (16,9010) ((FTAB(IF,JF),JF=1, 4),IF=1,NF)
 9010 FORMAT(I5,2A2,I5)
      READ (16,9011)((RFDIR(IR,JR),JR=1,  3),IR=1,NFR)
 9011 FORMAT(3I5)
      REWIND  16
C*******   READING  THE  RELATIONS  REQUIRED  BY  THE USER   BY NAMES **********
      DO 10I=1,NRQ,2
      DO 10 J =1,NR
      II=I+1
      IF((IREL(I).NE.RTAB(J,2)).OR.(IREL(II).NE.RTAB(J,3)))GO TO 10
      IP=IP+1
      IRID(IP)= RTAB(J,1)
 C *********   TESTING FOR CODE AUNTHORASIATION OF THE USER    ********************
      IF(ACCD(1).NE.RTAB(J,6)) GO TO 25
      LEVL=ACCD(2)
      IF(LEVL.LT.50)  IFLG=1
      GO TO (7,6),IFLG
    6 CONTINUE
 X    WRITE(6,8)
    8 FORMAT(20X,*USER---- YOU CAN     READ  ONLY FROM DATA BASE*)
      GO TO 11
    7 CONTINUE
 X    WRITE(6,9)
    9 FORMAT(20X,*USER---- YOU CAN     READ  WRITE FROM DB     *)
 C *********   IFLG IS 1 FOR READ ONLY  2 FOR BOTH READ AND WRITE  **************
   11 CONTINUE
 X    WRITE(6,1)IP,J
    1 FORMAT(10X,* ---- IREL  *,I3,*  MATCHED WITH*,I3,* IN RTAB*)
   10 CONTINUE
 X     WRITE(6,2) ACCD(1),IFLG
       WRITE(2,2) ACCD(1),IFLG
    2 FORMAT(10X,A2,10X,*AUTHORISED USER -----    LEVEL-----*,I2)
 X    WRITE(6,12)
 X 12 FORMAT(50X,*HERE ARE THE RELID AND REL NAMES*,//60X,*RELID*,5X,*RE
     1LNAME*)
 X    WRITE(6,13) ((RTAB(I,J),J=1,3),I=1,NR)
 X 13 FORMAT(60X,I3,9X,2A2)
       GO TO 30
   25 CONTINUE
 X    WRITE(6,3) ACCD(1)
      WRITE(2,3) ACCD(1)
    3 FORMAT(10X,A2,* ---- UNAUTHORISED   CODE   USED   REJECTED*)
      RETURN
C**********   HAVING TESTED AUTHORISATION PREPARING MATRIX MCFR  COMMON *********
C *********    BLANKING COMMON FIELDS RELATION  MATRIX MCFR BEFORE USE **********
   30 DO 40 L=1,10
      DO 40 M=1,10
   40 MCFR(L,M)=0
C ********   TAKING RELID AND TESTING FOR COMMON FIELD
      ICONT=0
      JCONT=0
      DO 90 IR= 1,NO
      DO 90 JR=1,NO
C ******MATCHING   FIRST     RELATIONS ID  WITH RFDIR          ********************
      DO 45 IFR=1,NFR
      IFF=IFR
      IF(IRID(IR).EQ.RFDIR(IFF,1)) GO TO 50
   45 CONTINUE
X     WRITE(6,4) IRID(IR)
      WRITE(2,4) IRID(IR)
    4 FORMAT(10X, I2,*----IRELID  NOT IN RFDIR  *)
      RETURN
```

```
C ******MATCHING   SECOND  * RELATIONS ID  WITH RFDIR          ********************
   50  DO 55 JFR=1,NFR
       JF=JFR
       IF(IRID(JR).EQ.RFDIR(JF,1) ) GO TO 60
   55  CONTINUE
 X     WRITE(6,5)  IRID(JR)
       WRITE(2,5)  IRID(JR)
   5   FORMAT(10X,I2,* -----JRELID   NOT IN   RFDIR  *)
       RETURN
C ********IRID OF BOTH I REL AND J REL HAVE BEEN MATCHEN    **********************
   60  IF(RFDIR(IFF,2).NE.RFDIR(JF,2)) GO TO 70
       MCFR(IR,JR)= RFDIR(IFF,2)
       MCFR(JR,IR)= RFDIR(IFF,2)
       GO TO 90
   70   JF=JF+1
       JCONT=JCONT+1
       IF(RFDIR(JF,1).EQ.IRID(JR)) GO TO 60
       ICONT=ICONT+1
       IFF=IFF+1
       JF=JF- JCONT
       IF(RFDIR(IFF,1).EQ.IRID(IR)) GO TO 60
       IFF=IFF-ICONT
   90  CONTINUE
       RETURN
       END


C
      SUBROUTINE BSERCH (KEY,ITAB,IUL,INDX)
C
C******************************************************************************
C*********** THIS SUBPROGRAM CARRIES OUT BINARY SEARCH IN ITAB WITH IUL *********
C*********  ENTRIES IN TABLE WITH KEY AND OUTPUTS INDW AS ENTRY IN ITAB**********
C     EXPLANATION OF VARIBLES USED FOLLOWS---------------
C     KEY      -----INTEGER KEY TO BE MATCHED
C     ITAB     -----NAME OF TABLE TO BE SEARCHED
C     IUL      -----UPPER LIMIT OO TOTAL NUMBER OF ENTRIES
C     INDX     -----OUTPUT LOCATION ON ITAB WHEN MATCH OCCURS
C*****************************************************************************DMD00030
      DIMENSION ITAB(20,10)
      LL=1
      IU=IUL
      INDX=0
      IF((KEY.LT.0).OR.(KEY.GT.IUL)) GO TO 11
   1  I=(LL+IU)/2
      IF(IU.LT.LL) GO TO 12
C****** COMPARING KEY WITH MID POINT IN RTAB                  *********************
      IF(KEY-ITAB(I,1))2,10,3
   2  IU=I-1
      GO TO 1
   3  LL=I+1
      GO TO 1
  10  INDX=I
```

```
      RETURN
   11 CONTINUE
X     WRITE(6,21)
      WRITE(2,21)
   21 FORMAT(10X,*  ERROR ---------BINARRY SEARCH NOT VALID KEY RANGE*)
      RETURN
   12 CONTINUE
X     WRITE(6,22) KEY
      WRITE(2,22) KEY
   22 FORMAT(10X,*  KEY  *,I5, *NOT IN TABLE*)
      RETURN
      END
      SUBROUTINE     CLOSE(IABC)
C
C***************************************************************************
C******  ONCE USER COMPLETES  USE OF DATA BASE  HE SHOUL CLOSE  IT  *********
C     EXPLANATION OF VARIBLES USED FOLLO S---------------
C     IABC    -----USER ACCESS CODE AND LEVEL TO TESTED
C***************************************************************************
      INTEGER RELID,FLDID,RFDIR,RTAB,FTAB,DNO,BUFF,ACCD,DA,CONT,DBA
      DIMENSION KARD(40),FTAB( 40,4),RFDIR(50,3),RTAB(20,10),BUFF( 2,40)
     1,IREL(20),IRID(10),ACCD(2),MCFR(10,10)YIFID( 0),IUSR(40,40),DBA(2)
      DIMENSION IABC(2)
      COMMON KARD,NR,NF,NI,NA,FTAB,FLDID,RTAB,RELID,L,NDC,CONT,DNO,NOR
     1 ,NFR,BUFF,NRQ,IREL,IRID,IFLG,MCFR,RFDIR,ACCD ,DA,NAT,NII,NN,IFID
     2 ,IUSR,ITFLG,DBA
X     WRITE(6,1001)
      WRITE(2,1001)
 1001 FORMAT(30X,*ENTERING  SUBROUTINE------CLOSE*)
C*********  TESTING FOR ACCES DCODE AND LEVEL OF AUTHETICATION   ***********
      IF((IABC(1).EQ.ACCD(1)).AND.(IABC(2).EQ.ACCD(2)))GO TO 10
X     WRITE(6,1) IABC(1),IABC(2)
      WRITE(2,1) IABC(1),IABC(2)
    1 FORMAT(10X,*UNAUTHORISED CLOSE CALL -------ACCESS CODE--*,A2,I2)
      RETURN
C********HAVING VARIFIED AUTHENTICITY OF USER CLOSE OPERATIVE FROM HERE*********
   10 NRQ=0
      ACCD(1)=8224
      ACCD(2)=0
      DO 20 I=1,10
      IRID(I)=0
      IFID(I)=0
      DO 20 J=1,10
      MCFR(I,J)=0
   20 CONTINUE
      DO 30 K=1,40
      DO 30 M=1,40
      IUSR(K,M)=8224
   30 CONTINUE
X     WRITE(6,2) IABC(1),IABC(2)
      WRITE(2,2) IABC(1),IABC(2)
    2 FORMAT(10X,*USER ACCES CODE--*,A2,*LEVEL---*,I2,*HAS COMPLETED
     1 USE OF DATA BASE *,//10X,60(1H-))
      RETURN
      END
      SUBROUTINE GETO(IREID,IRO,JR)
C*****************************************************************************#
C     THIS SUBROUTINE GETS/OUTPUTS IN USER AREA IUSR THE RECORD
C     OCCURRENCE REQUIRED BY THE USER
C     DESCRIPATION OF PARAMETERS FOLLOWS-----
C     IREID    -----RELATION ID
C     IRO      -----RECORD OCCURANCE OR TUPLE NUMBER REQUIRED
C     JR       -----FLAG IF 0  DNO NOT REWOUND IF1 IT REWINDS AFTER USE
C******************************************************************************
      INTEGER RELID,FLDID,RFDIR,RTAB,FTAB,DNO,BUFF,ACCD,DA,CONT,DBA
      DIMENSION KARD(40),FTAB( 40,4),RFDIR(50,3),RTAB(20,10),BUFF( 2,40)
     1,IREL(20),IRID(10),ACCD(2),MCFR(10,10),IFID(10),IUSR(40,40),DBA(2)
      COMMON KARD,NR,NF,NI,NA,FTAB,FLDID,RTAB,RELID,L,NDC,CONT,DNO,NOR
     1 ,NFR,BUFF,NRO,IREL,IRID,IFLG,MCFR,RFDIR,ACCD ,DA,NAT,NII,NN,IFID
     2 ,IUSR,ITFLG,DBA
      CONT=1
      NOR=1
C********       FINDING  DNO  FROM RTAB   WITH  GIVEN  RELID     ****************
      CALL BSERCH(IREID,RTAB,NR,IR)
      DNO=RTAB(IR,4)
C******   MATCHING   THE  FIRST  DNO  RELATION                 ******************
 5    DO 110  IK= 1, NR
      IRT=RTAB(IK,4)
      IF(DNO.EQ.IRT) GO TO 20
 110  CONTINUE
      WRITE(2,501)DNO
 501  FORMAT(10X,I2,* ----DNO NOT MATCHED ! *)
      RETURN
C********     TESTING  FOR  DESIRED    RELATION  ID              **************
 20   IF (RTAB(IR,1).EQ. RTAB(IK,1)) GO TO 50
      NDC=RTAB(IK, 7)
      NI =RTAB(IK, 8)
      NA =RTAB(IK, 9)
      DA =RTAB(IK,10)
 25   DO 30 J=1,NDC
      CALL IOPT
 30   CONTINUE
      IOFL=NDC/10+1
      DO 31 K=1,IOFL
      CALL IOPT
 31   CONTINUE
      IK=IK+1
      GO TO 20
C****************   HAVING  REACHED  THE BEGINING OF RELATION GO IOR ************
 50   NI   =RTAB(IR,8)
      NA=RTAB(IR,9)
      DA=RTAB(IR,10)
      NDC  =RTAB(IR,7)
      DO 55 K=1,IRO
      CALL IOPT
 55   CONTINUE
      IF(JR.LE.0) GO TO 60
C******* READING COMPLETED REWINDING THE APPROPRIATE DEVICE   *****************
      GO TO (62,63,64,65,66),DNO
```

```
   62  GO TO 75
   63   REWIND 13
       GO TO 75
   64  REWIND 14
       GO TO 75
   65  REWIND  15
       GO TO 75
   66  REWIND 16
   75  N =1
   60  DO 56 M=1,NN
   56  IUSR(1,M)=BUFF(1,M)
       N=1
  98   CONTINUE
 X     WRITE(6,99) IREID,IRO
  99   FORMAT(20X,*GETO  OUTPUT RELID-*,I5,*  IOR--*,I3,*FOLLOWS*)
                     WRITE(6,100)NI,(IUSR(N   ,M),M=1,NI),NAT,(IUSR(N   ,M),
     1M=NII,NN)
  100  FORMAT(NI5,NA2)
       RETURN
       END


       SUBROUTINE GETF(RID,FID,RO,FPOSN,FVR,FORT)
C**************************************************************************************
C      THIS SUBROUTINE GETS ASPECIFIED FIELD/ITEM VALUE AND OUTPUTS IT
C      POSITION(IN FPOSN),VALUE(IN FVR AND IUSR) AND FORMAT (IN FORTR
C      EXPLANATION OF VARIBLES USED FOLLOWS---------------
C      RID        -----THE RELATION ID OF THE DESIRED RELATION
C      FID        -----THE FIELD ID OF THE FIELD REQUIRED BY THE USER
C      RO         -----THE RECORD OCCURRENCE OF THE REALATION
C      FPOSN      -----THE POSITION OF THE REQUIRED FIELD IN THE OCCURRENCE
C      FVR        -----THE ARRAY OF FIELD VALUE REQUIRED
C      FORT       -----THE FORMAT OF THE REQUIRED
C**************************************************************************************
       INTEGER RELID,FLDID,RFDIR,RTAB,FTAB,DNO,BUFF,ACCD,DA,CONT,DBA
       INTEGER RID,FID,FPOSN,FVR,RO,FCONT,FORT
       DIMENSION KARD(40),FTAB( 40,4),RFDIR(50,3),RTAB(20,10),BUFF( 2,40)
      1,IREL(20),IRID(10),ACCD(2),MCFR(10,10),IFID(10),IUSR(40,40),DBA(2)
        DIMENSION FVR(30)
       COMMON KARD,NR,NF,NI,NA,FTAB,FLDID,RTAB,RELID,L,NDC,CONT,DNO,NOR
      1 ,NFR,BUFF,NRO,IREL,IRID,IFLG,MCFR,RFDIR,ACCD ,DA,NAT,NII,NN,IFID
      2 ,IUSR,ITFLG,DBA
C********  THIS SUBROUTINE FINDS VALUE OF A FIELD WHEN RELID,KFID,FOT***********
C************  NOR           IS  GIVEN                                *************
X      WRITE(6,1001)
 1001 FORMAT(10X,*ENTERING  SU ROUTINE---------  GETF*)
C*******  FINDING POSOTION  FID  AND FORT
      K=1
      FPOSN=1
      DO 10 I=1,NFR
      IT=I
      IF(RID.EQ.RFDIR(IT,1))GO TO 15
  10  CONTINUE
```

```
X     WRITE(6,1)  RID
      WRITE(2,1)  RID
  1   FORMAT(20X,I5,*-----RID  NOT  IN RFDIR   *)
      RETURN
C******RID  HAVING BEEN MATCHED  MATCHING FOR    FID             *******************
 15   IF(FID.EQ.RFDIR(IT,2))GO TO 20
C**********    TESTING   FOR   IN   AN   AD    FORMATS        *******************
      IF(RFDIR(IT,3).EQ.1)FPOSN=FPOSN+ 1
      IF(RFDIR(IT,3).EQ.2)FPOSN=FPOSN+ 10
      IF(RFDIR(IT,3).EQ.3)FPOSN=FPOSN+ 30
      IT=IT+1
      IF(RID.EQ.RFDIR(IT,1))GO TO 15
X     WRITE(6,2) FID
      WRITE(2,2) FID
  2   FORMAT(20X,I5,*-------FID  NOT  IN  RFDIR  *)
      RETURN
C*********  HAVING FOUND POSITION OF FID FINDING PHYSICAL LOC FROM RTAB*********
 20   FORT=      RFDIR(IT,3)
C*********  BRINGING  RO OCCURANCE  OF RECORD INTO  BUFFER        ***************
      CALL GETO(RID,RO,K)
C******* OUTPUTTING ACCORDING TO TYPE OF FORMAT    IN  OR  AN/AD    *************
X     WRITE(6,1306)RID,FID,FPOSN,FORT
 1306 FORMAT(10X,*RID--*,I3,*  FID  *,I3,*  FPOSN--*,I3,* FOT--*,I3)
      IF(FORT.GE.2)GO TO 30
      FVR(1)=IUSR(1,FPOSN)
      WRITE(6,1307) FVR(1)
 1307 FORMAT(10X,*GETF OUTPUT---------  *,I5)
      RETURN
 30   IF(FORT.EQ.2) LNF=10
      IF(FORT.EQ.3 ) LNF=30
      DO 40 J=1,LNF
      K=J+FPOSN-1
 40   FVR(J)=IUSR(1,K)
X     WRITE(6,1308) ( FVR(K),K=1,LNF)
 1308 FORMAT(10X,30A2)
      RETURN
      END


      SUBROUTINE GETV(GRID,GFID,GFV,RFID,RFV,JFORT,ION)
C
C     ******************************************************************
C ******  THIS SUBROUTINE FINDS A VALUE OF REQUIRED FIELD GIVEN
C     RELATION ID ,A KNOWN FIELD ID AND ITD VALUE
C     EXPLANATION OF VARIBLES USED FOLLOWS---------------
C     GRID     -----RELATION ID OF THE DESIRED RELATION
C     GFID     -----FIELD ID WHOSE VALUE IS GIVEN
C     GFV      -----GIVEN VALUE OF THE FIELD/ITEM
C     RFID     -----FIELD ID OF THE REQUIRED FIELD
C     JFORT    -----FORMAT OF THE REQUIRED FIELD
C     ION      -----NUMBER OF MATCHING OUTPITS REQUIRED
C***********************************************************************
      INTEGER RELID,FLDID,RFDIR,RTAB,FTAB,DNO,BUFF,ACCD,DA,CONT,DBA
      INTEGER GRID,GFID,GFV,RFID,RFV,GFCONT,RFCONT,GFOT,RFOT,GFLAG,RFLAG
      DIMENSION KARD(40),FTAB( 40,4),RFDIR(50,3),RTAB(20,10),BUFF( 2,40)
     1,IREL(20),IRID(10),ACCD(2),MCFR(10,10),IFID(10),IUSR(40,40),DBA(2)
      DIMENSION GFV(30),RFV(30),ICV(30)
      COMMON KARD,NR,NF,NI,NA,FTAB,FLDID,RTAB,RELID,L,NDC,CONT,DNO,NOR
     1 ,NFR,BUFF,NRQ,IREL,IRID,IFLG,MCFR,RFDIR,ACCD ,DA,NAT,NII,NN,IFID
     2 ,IUSR,ITFLG,DBA
X     WRITE(6,1001)
 1001 FORMAT(30X,*ENTERING  SUBROUTINE-----GETV *)
      IBL2=8224
      IOR=1
      JOR=1
      IT=2
      CALL GETF(GRID,RFID,JOR,JPN,RFV,JFORT)
      IF(GFID.NE.RFID) GO TO 20
C*************   CASE WHEN GFID IS SAME AS RFID                    *********************
      GO TO (7,8,9),JFORT
    7 RFV(1)=GFV(1)
      IUSR(IT,1)=GFV(1)
X     WRITE(6,100) IUSR(IT,1)
      GO TO 85
    8 LNF=10
      GO TO 10
    9  LNF=30
   10 DO 15 M=1,LNF
       RFV(M)=GFV(M)
      IUSR(IT,M)=GFV(M)
   15 CONTINUE
X     WRITE( 6,101)  LNF,(IUSR(IT,JK),JK=1,LNF)
      GO TO 85
C***********    POSITIONING  READ / WRITE HEAD AT THE BEGINNING OF REL *********
   20 CALL GETF(GRID,GFID,IOR,IPN,ICV,IFORT)
C******** LOCATON FROM RTAB USING BINARY SEARCH                    ********************
      CALL BSERCH(GRID,RTAB,NR,LR)
      ITFLG=1
C**** SUB SERCH LOCATES THE POSITION AT LR IN RTAB          ************************
      NDC =RTAB(LR,7)
C********* GETTING EACH REC FROM DISK AND COMPARING WITH GFV        *************
      DO 90 IOR=1,NDC
C*****ASSUMING FIND  IN IOPT BRINGS 1  RECORDS IN TO BUFF *********************
      CALL GETO(GRID,IOR, IT)
C************    COMPARING THE GFV VALUE WITH THE FD VALUE OF EACH REC**********
      GO TO (50,60,70),IFORT
   50 IF(GFV(1).EQ.BUFF(1,IPN)) GO TO 77
      GO TO 90
   60 LNF=10
      GO TO 75
   70 LNF=30
C**********  MATCHING GIVEN FIELD VALUE IN IGF OF RELATION FOR AN/IN FOT********
   75 DO 76 J=1,LNF
      JK=IPN+J-1
      IF(GFV(J).NE.BUFF(1,JK))GO TO 90
   76 CONTINUE
   77 GO TO (78,79,81),JFORT
   78 RFV(1)=BUFF(1,JPN)
```

```
      IUSR(IT,1)=BUFF(1,JPN)
C********   RFV  CONTAINS VALUE OF INTEGER FIELD                          *************
X     WRITE(6,100) RFV(1)
      GO TO 85
C*****IN CASE OF AN FORMAT OUT PUT IS 20/60 CHARACTERS IN ARRAY  **************
  79  LNF=10
      GO TO 82
  81  LNF=30
  82  DO 83 M=1,LNF
      MM=JPN+M-1
      IUSR(IT,M)=BUFF(1,MM)
  83  RFV(M)=BUFF(1,MM)
X     WRITE(6,101) LNF,(RFV(JJ),JJ=1,LNF)
  85  CONTINUE
X     WRITE(6,102) GRID,RFID,JFORT
 102  FORMAT(5X,*GETV- OP--RID*,I5,*  FID*,I3,*FOT*,I3,*ABOVE*)
      IT=IT+1
       IF(IT.GT.ION)  RETURN
  90  CONTINUE
      ITFLG=0
 100  FORMAT(10X,I6)
 101  FORMAT(10X,NA2)
      RETURN
      END


      SUBROUTINE GETQ(QRID,QFID,QOPR,QFV,ION)
C
C*******************************************************************************
C     THIS SUBROUTINE OUTPUTSDIFFERENT FIELDS/ITEMS FROM DIFFERENT
C      RELATIONS GIVEN IN IRID AND IFID WHICH SATISIFY A PARTICULAR
C     QUALIFICATION
C     EXPLANATION OF VARIBLES USED FOLLOWS---------------
C     QRID     -----THE QUALIFIED RELATION ID
C     QFID     -----THEQUALIFING FIELD ID
C     QOPR     -----THE CONDITIONAL OPERATOPR
C**********      LT=1 LE=2 EQ=3  GE4  GT=5  GT=6    CODES FOR QOPR    ***********
C     QFV      -----THE VALUE OF THE FIELD TO BE MATCHED
C     ION      -----THE NUMBER OF OUTPUT REQUIRED WHICH MATCH
C*******************************************************************************
      INTEGER RELID,FLDID,RFDIR,RTAB,FTAB,DNO,BUFF,ACCD,DA,CONT,DBA
      INTEGER QRID,QFID,QOPR,QFV
      DIMENSION KARD(40),FTAB( 40,4),RFDIR(50,3),RTAB(20,10),BUFF( 2,40)
     1,IREL(20),IRID(10),ACCD(2),MCFR(10,10),IFID(10),IUSR(40,40),DBA(2)
      DIMENSION IFEL(10),QFV(30),IRFV(30),ICFV(30),              IGFV(30)
      COMMON KARD,NR,NF,NI,NA,FTAB,FLDID,RTAB,RELID,L,NDC,CONT,DNO,NOR
     1 ,NFR,BUFF,NRQ,IREL,IRID,IFLG,MCFR,RFDIR,ACCD ,DA,NAT,NII,NN,IFID
     2 ,IUSR,ITFLG,DBA
X     WRITE(6,1001)
 1001 FORMAT(30X,*ENTERING  SUBROUTINE-----GETQ *)
C**********      INITIALISATION  OF  LOCAL  VARIABLES                    **************
      NQ=NRQ/2
X     WRITE(6,106) NQ,(IFID(J),J=1,NQ)
```

```
X     WRITE(6,105) NQ,(IRID(J),J=1,NQ)
      WRITE(2,106) NQ,(IFID(J),J=1,NQ)
      WRITE(2,105) NQ,(IRID(J),J=1,NQ)
  105 FORMAT(10X,*FROM RELATIONS ID--*,NI5)
  106 FORMAT(10X,*GETVALUES OF FLDIDS--*,NI5)
      LNF=0
      I=0
      J=1
      CALL BSERCH (QRID,RTAB,NR,IR)
      NDC=RTAB(IR,7)
C*********     STARTING THE LOOP FOR TESTING EACH FD IN QRID     ***************
      DO 11 IOR=1,NDC
      INOR=IOR
C********** FINDING POSITION  AND  FORMAT  OF QUAL  FIELD    ******************
      CALL GETF(QRID,QFID,IOR,IP,IRFV,IFORT )
C**********   TESTING ACCORDING TO QOPR USING COMPUTED GO TO    ***************
      GO TO (1,2,3,4,5,6), QOPR
   1  IF(IFORT.GT.1)  GO TO 99
      IF(IRFV(1).LT.QFV(1)) GO TO 10
      GO TO 11
   2  IF(IFORT.GT.1)  GO TO 99
      IF(IRFV(1).LE.QFV(1)) GO TO 10
      GO TO 11
   3  IF(IFORT.GT.1) GO TO 13
      IF(IRFV(1).EQ.QFV(1)) GO TO 10
      GO TO 11
  13  IF(IFORT.EQ.2) LNF=10
      IF(IFORT.EQ.3) LNF=30
      DO 14 IM=1,LNF
      IF(IRFV(I).NE.QFV(I)) GO TO 11
  14  CONTINUE
      GO TO 10
   4  IF(IFORT.GT.1)  GO TO 99
      IF(IRFV(1).GE.QFV(1)) GO TO 10
      GO TO 11
   5  IF(IFORT.GT.1)  GO TO 99
      IF(IRFV(1).GT.QFV(1)) GO TO 10
      GO TO 11
   6  IF(IFORT.GT.1)  GO TO 99
      IF(IRFV(1).NE.QFV(1)) GO TO 10
      GO TO 11
C********   HAVING OBTAINED MATCH IN QREL/FD VALUE OR SATISFIED OPREATTOR*******
C*********    TESTING FOR COMMON FIELFD WITH IREL AN D QRID          **************
  10  I=I+1
      IF(I.GT. NQ) GO TO 50
      NRID=IRID(I)
      NFID=IFID(I)
C**********   FINDING  COMMON FIELD  FROM  MCFR BETWEEN QFID AND NRID **********
      DO 15 IT=1,NQ
      IF(QRID.EQ.IRID(IT))IT1=IT
      IF(NRID.EQ.IRID(IT))IT2=IT
  15  CONTINUE
      ICFD=MCFR(IT1,IT2)
X     WRITE(6,107) IT1,IT2,ICFD
```

```
C*****************************************************************************************
       INTEGER RELID,FLDID,RFDIR,RTAB,FTAB,DNO,BUFF,ACCD,DA,CONT,DBA
       INTEGER FORT,FOT
       DIMENSION KARD(40),FTAB( 40,4),RFDIR(50,3),RTAB(20,10),BUFF( 2,40)
     1,IREL(20),IRID(10),ACCD(2),MCFR(10,10),IFID(10),IUSR(40,40),DBA(2)
       DIMENSION ITID(10),IMID(10),IGFV(30),IRFV(30),ICFV(30)
       COMMON KARD,NR,NF,NI,NA,FTAB,FLDID,RTAB,RELID,L,NDC,CONT,DNO,NOR
     1 ,NFR,BUFF,NRQ,IREL,IRID,IFLG,MCFR,RFDIR,ACCD ,DA,NAT,NII,NN,IFID
     2 ,IUSR,ITFLG,DBA
C********    INITIALISATION OF LOCAL  VARIABLE                       *******************
X      WRITE(6,1001)
 1001 FORMAT(30X,*ENTERING  SUBROUTINE------GETU *)
C********    TESTING  IF FIRST IFID IS PRIMARY  KEY OF FIRTS  RELATION**********
      LNF=0
      NQ=NRQ/2
      KR1=ITID(1)
      KF1=IMID(1)
C**********    FINDING NUMBER OF RECORDS IN FIRST  RELATION          **************
      CALL BSERCH(KR1,RTAB,NR,IR)
      NDC=RTAB(IR,7)
C*******      FINDING  PRIMARY  KEY OF FIRST  RELATION               ***************
      DO 10 I=1,NFR
      IF(RFDIR(I,1).EQ.KR1) GO TO 15
  10  CONTINUE
C*********  FIRST FIELD NOT BEING PRIMARY KEY FINDING THE SAME    **************
  15  KEYID=RFDIR(I,2)
C******LOCATED PRIMARY KEY OF FIRST RELATION OUTPUT VALUE IN IGFV  ***********
      DO 90 IRO=1,NDC
      IJ=1
      IQ=1
      IQ2=IQ+1
C**********        FINDING   KEY FIELD  VALUE FOR FIRST RELATION
      CALL GETF(KR1,KEYID,IRO,JPSN,IGFV,FORT)
C********  IGFV V  CONTAINS    KEY FD  VALUE                  *********************
      JFOT=FORT
C*********** IRFV  CONTAINS FIRST FIELD VALUE                        ****************
      IT=1
      CALL GETV(KR1,KEYID,IGFV,KF1,IRFV,FORT,IT)
       FOT=FORT
C******      FINDING  POSITION OF FIRST AND SECOND RELATION IN  MCFR*************
  20  ID1=ITID(IQ)
      KD1=IMID(IQ )
      IF(IQ2.GT.NQ) GO TO 28
  22  ID2=ITID(IQ2)
      KD2=IMID(IQ2)
C*******FINDING VALUE OF COMMON FIELD  FROM COMMON MATRIX           ***************
X     WRITE(6,104) (IMID(J),J=1,NQ),ID1,KD1,ID2,KD2
 104 FORMAT(  X,*VALUE OF IRIDS  *,10I5)
      DO 25 IP=1,NRQ
      IF(ID1.EQ.IRID(IP)) IP1=IP
      IF(ID2.EQ.IRID(IP)) IP2=IP
  25  CONTINUE
C********  FINDING  COMMON FIELDID WITH SECOND IRID
      ICFD=MCFR(IP1,IP2)
```

```
X     WRITE(6,26) ID1,ID2,ICFD
  26  FORMAT(20X,*RELID1--*,I2,*RELID2--*,I2,*COMMON FD--*,I3 )
C*******  ICFVFV  CONTAI  COMMON   FIELD   VALUE
      CALL GETV(ID1,KEYID,IGFV,ICFD,ICFV,FORT,IT)            **********************
      KFOT=FORT
C********      OUTPUTTING  FIRST   VALUE   DEPENDING  ON  FOT VALUE**************
  28  CONTINUE
      WRITE(2,103)ID1,KD1,IRO,FOT
X     WRITE(6,103)ID1,KD1,IRO,FOT
      IC=IJ
      GO TO (29,30,31),FOT
  29  IUSR(IRO,IJ)=IRFV(1)
X     WRITE(6,101)          (IUSR(IRO,IJ) )
      WRITE(2,101)          (IUSR(IRO,IJ) )
      IJ=IJ+1
      GO TO 50
  30  LNF=10
      GO TO 32
  31   LNF=30
  32  DO 40 J=1,LNF
      IUSR(IRO,IJ)=IRFV(J)
  40  IJ=IJ+1
C********    OUTPUTTING   DEPENDING  ON  VALUE  OF FOT        ******************
      WRITE(2,102) (IRFV(JK),JK=1,LNF )
X     WRITE(6,102) (IUSR(IRO,JK),JK=IC,LNF)
C*******FINDING  SECOND VALUE USING  ICFD                 **********************
  50  IQ=IQ+1
      IQ2=IQ+1
      IF(ITFLG.LT. 1) GO TO 60
      IF(IQ.GT.NQ) GO TO 90
C********* OUTPUTTING SECOND FIELD VALUE USING COMMON FIELD      ***************
  45  CALL GETV(ID2,ICFD,ICFV,KD2,IRFV,FORT,IT)
       FOT=FORT
      GO TO 28
  60  ID1=ITID(IQ)
      KD1=IMID(IQ)
       KEYID=ICFD
      ITFLG=1
      IF(IQ2.GT.NQ) GO TO 90
      GO TO 22
  90  CONTINUE
 101  FORMAT(50X,I5)
 102  FORMAT(20X,30A2)
 103  FORMAT(5X,*GETU OP--RID*,I5,*FID*,I3,* IRO*,I3,*FOT*,I2,*FOLLOW*)
      RETURN
      END
      SUBROUTINE GETR(IRI,IABC)
C*********************************************************************************
C**********  THIS SUBROUTINE OUTPUTS A COMPLETE RELATION RELID IS IRI ****************
C     EXPLANATION OF VARIBLES USED FOLLO S---------------
C     IRI     -----THE RELATION ID WHICH IS TO BE OUTPUTTED
C     IABC    -----USER ACCESS CODE AND LEVEL TO TESTED
C*********************************************************************************
      INTEGER RELID,FLDID,RFDIR,RTAB,FTAB,DNO,BUFF,ACCD,DA,CONT,DBA               DMD00050
      DIMENSION KARD(40),FTAB( 40,4),RFDIR(50,3),RTAB(20,10),BUFF( 2,40)
     1,IREL(20),IRID(10),ACCD(2),MCFR(10,10),IFID(10),IUSR(40,40),DBA(2)
      DIMENSION IABC(2)
      COMMON KARD,NR,NF,NI,NA,FTAB,FLDID,RTAB,RELID,L,NDC,CONT,DNO,NOR
     1 ,NFR,BUFF,NRQ,IREL,IRID,IFLG,MCFR,RFDIR,ACCD ,DA,NAT,NII,NN,IFID
     2 ,IUSR,ITFLG,DBA
X     WRITE(6,1001)
      WRITE(2,1001)
 1001 FORMAT(30X,*ENTERING  SUBROUTINE------GETRS*)
C*********     TESTING  USER ACCES   CODES    FOR AUTHORISATOIN  *************
      IF(ACCD(1).EQ.IABC(1)) GO TO 3
X     WRITE(6,4) IABC(1)
      WRITE(2,4) IABC(1)
  4   FORMAT(10X,A2, *----ACCES CODE NOT MATCHEDBY USER *)
      RETURN
C********  TSETING  USER  ACCES  LEVEL FOR  USING  ADDREL CALL  *********M*****
  3   IF(IABC(2).LT.10) GO TO 7
X     WRITE(6,8) IABC(2)
      WRITE(2,8) IABC(2)
  8   FORMAT(10X,I2,*LEVEL GREATER THAN 10 USER NOT ALLOWED  ADDREL *)
      IFLG=2
      RETURN
C*******  HAVING SATISIFIED USER AUTHORISATION PROCEEDING        ************
  7   NOR=1
      CALL BSERCH (IRI ,RTAB,NR,IR)
      DNO=RTAB(IR, 4)
      IL =RTAB(IR, 5)
      NDC=RTAB(IR, 7)
      NA =RTAB(IR, 8)
      NI =RTAB(IR, 9)
      DA =RTAB(IR,10)
C       REACHING THE BEGINING OF ARELATION USING GETO
      JO=0
      IOR=1
      ID=IRI
      CALL GETO(ID,IOR,JO)
X     WRITE( 6,100)NI,(IUSR(1,J),J=1,NI),NAT,(IUSR(1,J),J=NII,NN)
      DO110 I=1,NDC
      CONT=1
      L=I
      CALL IOPT
C********   OUTPUTTING  ON  PRINTER   AND  IN  USER  AREA     **************
      DO 20 JK=1,NN
  20  IUSR(L,JK)=BUFF(1,JK)
X     WRITE( 6,100)NI,(IUSR(I,J),J=1,NI),NAT,(IUSR(I,J),J=NII,NN)
 110  CONTINUE
```

```
C     REWINDING THE DEVICE AS PER DNO USINS COMPUTED GO TO
      GO TO (6,13,14,15,16),DNO
  6   RETURN
 13   REWIND  13
      RETURN
 14   REWIND  14
      RETURN
 15   REWIND  15
      RETURN
 16   REWIND  16
100   FORMAT(NI5,NA2)
      RETURN
      END


      SUBROUTINE DUMP(IABC)
C*****************************************************************************************
C******* ON REQUEST FROM USER DUMP ROUTINE PROVIDES ALL TABLES AND RELAS********
C     EXPLANATION OF VARIBLES USED FOLLO S---------------
C     EXPLANATION OF VARIBLES USED FOLLO S---------------
C     IABC    -----USER ACCESS CODE AND LEVEL TO TESTED
      INTEGER RELID,FLDID,RFDIR,RTAB,FTAB,DNO,BUFF,ACCD,DA,CONT,DBA
      DIMENSION KARD(40),FTAB( 40,4),RFDIR(50,3),RTAB(20,10),BUFF( 2,40)
     1,IREL(20),IRID(10),ACCD(2),MCFR(10,10),IFID(10),IUSR(40,40),DBA(2)
      DIMENSION IABC(2)
      COMMON KARD,NR,NF,NI,NA,FTAB,FLDID,RTAB,RELID,L,NDC,CONT,DNO,NOR
     1 ,NFR,BUFF,NRG,IREL,IRID,IFLG,MCFR,RFDIR,ACCD ,DA,NAT,NII,NN,IFID
     2 ,IUSR,ITFLG,DBA
X     WRITE(6,1001)
      WRITE(2,1001)
1001  FORMAT(30X,*ENTERING  SUBROUTINE------ DUMP*)
C********     TESTING  USER ACCES   CODES    FOR AUTHORISATOIN  **************
      IF(ACCD(1).EQ.IABC(1)) GO TO 3
X     WRITE(6,4) IABC(1)
      WRITE(2,4) IABC(1)
  4   FORMAT(10X,A2, *----ACCES CODE NOT MATCHEDBY USER *)
      RETURN
C********  TSETING  USER  ACCES  LEVEL FOR  USING  ADDREL CALL  *********M*****
  3   IF(IABC(2).LT.10) GO TO 7
X     WRITE(6,8) IABC(2)
      WRITE(2,8) IABC(2)
  8   FORMAT(10X,I2,*LEVEL GREATER THAN 10 USER NOT ALLOWED  ADDREL *)
      IFLG=2
      RETURN
C*******   HAVING SATISIFIED USER AUTHORISATION PROCEEDING         ************
  7   NQ=NRQ/2
X     WRITE(6,2)ACCD(1),ACCD(2)
      WRITE(2,2)ACCD(1),ACCD(2)
  2   FORMAT(10X,*USER ACCES CODE--*,A2,*LEVEL---*,I2,*REQUESTS FOR
     1DUMP*)
X 20  WRITE(6,25)
X 25  FORMAT(1H1,40X, * A L L  R E L A T I O N S  I N  D A T A B A S E
     1 *,/40X,60(1H-))
```

```
C***********************************************************************
X      WRITE( 6, 1006)
 1006 FORMAT(10X,*CONTENTS  OF  RTAB *,//, 5X,*RELID NAME DNOSF AC NDC
     1 NA   NI   DA   *)
X      WRITE(6,1009) ((RTAB(I,J),J=1,10),I=1,NR)
X      WRITE(6,1111)      NR
 1009 FORMAT(  X,I5,2A2,2I5,A2,4I5)
X      WRITE( 6, 1007)
 1007 FORMAT(5 X,*CONTENTS  OF  FTAB *,//5 X,*FLDID      NAME    FORMAT*)
X       WRITE(6,1010) ((FTAB(IF,JF),JF=1, 4),IF=1,NF)
X      WRITE(6,1111)      NF
 1010 FORMAT(5 X,I5,5X,2A2,5X,I5)
X      WRITE( 6, 1008)
 1008 FORMAT(10X,*CONTENTS  OF  RFDIR*,//,10X,*RELID       FLDID    FORMA
     1T*)
X      WRITE(6,1011) ((RFDIR(IR,JR),JR=1,  3),IR=1,NFR)
 1011 FORMAT(10X,I5,10X,I5,10X,I5)
 1111 FORMAT(50X,*NUMBER OF ENTRIES IN TABLE--*,I5)
      DO 10 I=1,NQ
      ID=IRID(I)
      WRITE(2,5) ID
    5 FORMAT(10X,*USER REQUIRED RELID--*,I5)
      CALL GETR(ID,IABC)
   10 CONTINUE
X      WRITE(6,30)
      WRITE(2,30)
X      WRITE(6,5) ID
   30 FORMAT(10X,* USER DUMP    C O M P L E T E D *,/,10X,60(1H-))
      RETURN
      END
      SUBROUTINE ADDREC(RID,IV,IABC)
C
C*****************************************************************************
C********   THIS SUBROUTINE ADDS A NEW RECORD IN A RELATION DURING RUN TIME**
C      EXPLANATION OF VARIBLES USED FOLLOWS---------------
C      RID      -----RELATION ID OF THE RELATION IN WHICH REC IS ADDED
C      IV       -----ARRAY HOLDING THE VALUE OF RECODRD TO BE ADDED
C      IABC     -----USER ACCESS CODE  FOR USING THE PRIVILAGED ROUTINE
C*****************************************************************************
      INTEGER RELID,FLDID,RFDIR,RTAB,FTAB,DNO,BUFF,ACCD,DA,CONT,DBA
      INTEGER RID
      DIMENSION KARD(40),FTAB( 40,4),RFDIR(50,3),RTAB(20,10),BUFF( 2,40)
     1,IREL(20),IRID(10),ACCD(2),MCFR(10,10),IFID(10),IUSR(40,40),DBA(2)
      DIMENSION IV(40) ,IABC(2)
      COMMON KARD,NR,NF,NI,NA,FTAB,FLDID,RTAB,RELID,L,NDC,CONT,DNO,NOR
     1 ,NFR,BUFF,NRQ,IREL,IRID,IFLG,MCFR,RFDIR,ACCD ,DA,NAT,NII,NN,IFID
     2 ,IUSR,ITFLG,DBA
X     WRITE(6,1001)
      WRITE(2,1001)
 1001 FORMAT(30X,*ENTERING  SUBROUTINE----ADDREC*)
C*******        INITIALISATION   OF   VARIABLES        ******************
      NOR=1
C*********      TESTING  USER ACCES   CODES    FOR AUTHORISATOIN  **********
      IF(ACCD(1).EQ.IABC(1)) GO TO 100
X     WRITE(6,5) IABC(1)
      WRITE(2,5) IABC(1)
   5  FORMAT(10X,A2, *----ACCES CODE NOT MATCHEDBY USER *)
      RETURN
C********  TSETING  USER  ACCES  LEVEL FOR  USING  ADDREL CALL  *********M****
 100  IF(IABC(2).LT.10) GO TO115
X     WRITE(6,6) IABC(2)
      WRITE(2,6) IABC(2)
   6  FORMAT(10X,I2,*LEVEL GREATER THAN 10 USER NOT ALLOWED  ADDREL *)
      IFLG=2
      RETURN
C***** HAVING VARIFIED USER AUTHORISATION MODIFYING THE RECORD**************
C******   MATCHING  RID IN  RTAB      ***********************************
 115  CALL BSERCH(RID,RTAB,NR,IR)
      DNO=RTAB(IR, 4 )
      ISP=RTAB(IR,5)
      NDC=RTAB(IR,7 )
      NI=RTAB(IR, 8)
      NII=NI+1
      NA=RTAB(IR, 9)
      DA=RTAB(IR,10)
C********      TESTING IF NDC IS EQUAL TO SP OF NEXT RELATION    *************
      IF(IR-(IR/6*6)) 30,30,20
  20  IRI=IR+1
      ISPN=RTAB(IRI,5)
      ITSP=ISP+NDC
      IF(ITSP.GE.ISPN)GO TO99
C     GETTING THE FIRST EMPTY SPACE AT THE END OF RELATION
  30  JO=0
      IO=NDC
      CALL GETO(RID,IO,JO)
      WRITING NEW RECORD INTO BUFFER AFTER UPDATING RTAB
      NOR=1
      NDC=NDC+1
      RTAB(IR,7)=NDC
      IF(NI.LT.1) GO TO 55
      DO 51K =1,NI
      BUFF(NOR,K)=IV(K)
  51  CONTINUE
C******    WRITING   IN  NA/DA  FORMAT                         ***************
  55  IF(NAT.EQ.0) GO TO 65
      DO 60  K=NII,NAT
  60  BUFF(NOR,K)=IV(K)
C******* HAVING WRITTEN ON THE NEW REC IN BUFFBUFF WRITING ON DISK**************
  65  CONT=2
      CALL IOPT
C******** REWINDING  THE PROPER DEVICE FOR FUTURE USE     ********************
      GO TO (61,62,63,64,66),DNO
  61  RETURN
  62  REWIND 13
      RETURN
  63  REWIND 14
      RETURN
  64  REWIND 15
      RETURN
  66  REWIND 16
      RETURN
  99  CONTINUE
      WRITE(6,102)
      WRITE(2,102)
 102  FORMAT(5X,*OVERFLOW AREA FULL RELOAD DMD FOR ADDING REC *)
      RETURN
      END


       SUBROUTINE MDYREC(RID,FID,IOR,IC,IABC)
C*****************************************************************************
C******* THIS SUBROUTINE MODIFIES A PARTICULAR FD VALUE WITH GIVEN VALUE*******
C      EXPLANATION OF VARIBLES USED FOLLOWS---------------
C      RID       -----RELATION ID OF THE RELATION TO BE MODIFIED
C      FID       -----FIELD ID OF THE FIELD TO BE MODIFIED
C      IOR       -----RECORD OCCURRENCE TO BE MODIFIED
C      IC        -----ARRAY HOLDING NEW VALUE
C      IABC      -----USER ACCESS CODE  FOR USING THE PRIVILAGED ROUTINE
C****************************************************************************
       INTEGER RELID,FLDID,RFDIR,RTAB,FTAB,DNO,BUFF,ACCD,DA,CONT,DBA
       INTEGER RID,FID,FPOSN
       DIMENSION KARD(40),FTAB( 40,4),RFDIR(50,3),RTAB(20,10),BUFF( 2,40)
      1,IREL(20),IRID(10),ACCD(2),MCFR(10,10),IFID(10),IUSR(40,40),DBA(2)
       DIMENSION IC(30),IV(30),IABC(2)
       COMMON KARD,NR,NF,NI,NA,FTAB,FLDID,RTAB,RELID,L,NDC,CONT,DNO,NOR
      1 ,NFR,BUFF,NRG,IREL,IRID,IFLG,MCFR,RFDIR,ACCD ,DA,NAT,NII,NN,IFID
      2 ,IUSR,ITFLG,DBA
```

```
X      WRITE(6,1001)
       WRITE(2,1001)
 1001 FORMAT(30X,*ENTERING  SUBROUTINE----MDYREC  *)
C********    TESTING  USER ACCES   CODES    FOR AUTHORISATOIN   **************
       IF(ACCD(1).EQ.IABC(1)) GO TO 101
X      WRITE(6,5) IABC(1)
       WRITE(2,5) IABC(1)
  5    FORMAT(10X,A2, *----ACCES CODE NOT MATCHEDBY USER *)
       RETURN
C********  TSETING  USER  ACCES  LEVEL FOR  USING  ADDREL CALL  ********M*****
 101   IF(IABC(2).LT.10) GO TO115
X      WRITE(6,6) IABC(2)
       WRITE(2,6) IABC(2)
  6    FORMAT(10X,I2,*LEVEL GREATER THAN 10 USER NOT ALLOWED  ADDREL *)
       IFLG=2
       RETURN
C*******   HAVING SATISIFIED USER AUTHORISATION PROCEEDING         ***********
 115   CALL GETF (RID,FID,IOR,FPOSN,IV,IFOT)
             GO TO (1,2,3),IFOT
   1   IF(IC(1).EQ.IV(1)) GO TO 98
       GO TO   11
   2   LNF=10
       GO TO    4
   3   LNF=30
   4   DO 10 I=1,LNF
       IF(IC(I).NE.IV(I)) GO TO 11
  10   CONTINUE
  98   WRITE(6,99)
  99   FORMAT(10X,* NEW AND OLD VALUES SAME-- NO ACTION REQUIRED*)
       RETURN
C********    HAVING TESTED VALUE WRITING MODIFYING FD VALUE        ************
  11   JO=1
       CALL  GETO(RID, IOR, JO)
       DO 12  IP=1,40
  12   KARD(IP)=IUSR(1,IP)
       WRITE(2,100)NI,(KARD(IS),IS=1,NI),NAT,(KARD(IS),IS=NII,NN)
       GO TO (21,22,23),IFOT
  21   KARD(FPOSN)=IC(1)
       GO TO 60
  22   LNF=10
       GO TO 55
  23   LNF=30
  55   DO 50K=1,LNF
       KARD(FPOSN)=IC(K)
       FPOSN=FPOSN+1
  50   CONTINUE
  60   JO=0
       IOR=IOR-1
       CALL GETO(RID, IOR, JO)
       DO 13 JP=1,40
  13   BUFF(1,JP)=KARD(JP)
       WRITE(2,100)NI,(KARD(IS),IS=1,NI),NAT,(KARD(IS),IS=NII,NN)
 100   FORMAT(NI5,NA2)
```

```
C      HAVING MODIFIED IN BUFFER  WRITING ON THE DISK
      CONT=2
      IOR=IOR+1
      NOR=1
      CALL IOPT
C******** REWINDING  THE PROPER DEVICE FOR FUTURE USE    ********************
      GO TO (61,62,63,64,66),DNO
   61 RETURN
   62 REWIND 13
      RETURN
   63 REWIND 14
      RETURN
   64 REWIND 15
      RETURN
   66 REWIND 16
      RETURN
      END


      SUBROUTINE ADDREL(RNAM,INDC,INAM,INI,INA,IDA,IABC)
C
C************************************************************************
C     THIS SUBROUTINE ADDS A NEW RELATION DURING RUN TIME
C     EXPLANATION OF VARIBLES USED FOLLOWS---------------
C     RNAM     -----RELATION NAME TO BE ADDED
C     INDC     -----NUMBER OF DATA CARDS OR REC OCCURRENCES
C     INAM     -----N ARRAY CONTAINING FIELD/ITEM NAMES TO BE ADDED
C     INI      -----NUMBER OF IN   FIELDS
C     INA      -----NUMBER OF AN   FIELDS
C     IDA      -----NUMBER OF DA   FIELDS
C     IABC     -----USER ACCESS CODE  FOR USING THE PRIVILAGED ROUTINE
C************************************************************************
      INTEGER RELID,FLDID,RFDIR,RTAB,FTAB,DNO,BUFF,ACCD,DA,CONT,DBA
      INTEGER RNAM(2)
      DIMENSION KARD(40),FTAB( 40,4),RFDIR(50,3),RTAB(20,10),BUFF( 2,40)
     1,IREL(20),IRID(10),ACCD(2),MCFR(10,10),IFID(10),IUSR(40,40),DBA(2)
      DIMENSION IABC(2),INAM(20)
      COMMON KARD,NR,NF,NI,NA,FTAB,FLDID,RTAB,RELID,L,NDC,CONT,DNO,NOR
     1 ,NFR,BUFF,NRQ,IREL,IRID,IFLG,MCFR,RFDIR,ACCD ,DA,NAT,NII,NN,IFID
     2 ,IUSR,ITFLG,DBA
X     WRITE(6,1001)
      WRITE(2,1001)
 1001 FORMAT(30X,*ENTERING  SUBROUTINE----ADDREL  *)
C*********      TESTING  USER ACCES   CODES    FOR AUTHORISATOIN ***************
      IF(ACCD( 1 ) .EQ.IABC(1) ) GO TO 10
X     WRITE(6,1) IABC(1)
      WRITE(2,1) IABC(1)
    1 FORMAT(10X,A2, *----ACCES CODE NOT MATCHEDBY USER *)
      IFLG=2
      RETURN
C********* TSETING  USER  ACCES  LEVEL FOR  USING  ADDREL CALL  *********M*****
   10 IF(IABC(2).LT.10) GO TO115
X     WRITE(6,2) IABC(2)
```

```
         2,2) IABC(2)
         (10X,I2,*LEVEL GREATER THAN 10 USER NOT ALLOWED  ADDREL *)
         IAVING SATISIFIED USER AUTHORISATION PROCEEDING TO DELET REL*******

      LIC=0
      NOR=1
      CALL BSERCH(IL,RTAB,NR,IR)
      IF(IR.EQ.0) GO TO 120
      DNO=RTAB(IR,4)
      L  =RTAB(IR,5)
      NDC=RTAB(IR,7)
      NI =RTAB(IR,8)
      NA=RTAB(IR,9)
      DA=RTAB(IR,10)
      IF((NDC.NE.INDC).OR.(NI.NE.INI).OR.(NA.NE.INA).OR.(DA.NE.IDA))GO
     1 TO 120
      LIC=1
      INR=NR
      RTAB(NR,1)=IR
      IOR=1
      GO TO 125
 120  DNO=RTAB(NR,4)
 125  IOR=RTAB(NR,7)+RTAB(NR,7)/10+1
      IF(LIC.EQ.1) IOR=1
      JO=0
      CALL GETO(NR,IOR,JO)
      NI=INI
      NA=INA
      DA=IDA
      NII=NI+1
      NAT=10*NA+30*DA
      NN=NI+NAT
      DO130 J=1,INDC
      NOR=1
      DO 30 K=1,NN
      JJ=J+1
  30  BUFF(1,K)=IUSR(JJ,K)
       CONT=2
      CALL  IOPT
 130  CONTINUE
C******** REWINDING  THE PROPER DEVICE FOR FUTURE USE    ********************
      GO TO (61,62,63,64,66),DNO
  61  GO TO 40
  62  REWIND 13
      GO TO 40
  63  REWIND 14
      GO TO 40
  64  REWIND 15
      GO TO 40
  66  REWIND 16
```

```
C********    HAVING   WRITTEN  ON DISK  UPDATING  THE SYSTEM TABLES **********
   40   IF(LIC.NE.1) NR=NR+1
        RTAB(NR,1)=NR
        RTAB(NR,2)=RNAM(1)
        RTAB(NR,3)=RNAM(2)
        RTAB(NR,4)=DNO
        JNR=NR-1
        IF((JNR.EQ.0).OR.(JNR.EQ.6).OR.(JNR.EQ.12))GO TO 41
        L=RTAB(JNR,5)
        RTAB(NR,5)=L+RTAB(JNR,7)/10+1
        GO TO 42
   41   RTAB(NR,5)=1
   42   RTAB(NR,6)=IABC(1)
        RTAB(NR, 7)=INDC
        RTAB(NR, 8)=INI
        RTAB(NR, 9)=INA
        RTAB(NR,10)=IDA
        IF(INI.EQ.0) GO TO 70
        N=NI
        IFOT=1
        M=0
   45   DO 60 I=1,N
        M=M+1
        MM=2*M
        K=MM-1
   46   DO 50 J=1,NF
   50   IF((INAM(K).EQ.FTAB(J,2)).AND.(INAM(MM).EQ.FTAB(J,3))) GO TO 55
        NF=NF+1
        FTAB(NF,1)=NF
        FTAB(NF,2)=INAM(K)
        FTAB(NF,3)=INAM(MM)
        FTAB(NF,4)=IFOT
        NFR=NFR+1
        RFDIR(NFR,1)=NR
        RFDIR(NFR,2)=NF
        RFDIR(NFR,3)=IFOT
        GO TO 60
   55   NFR=NFR+1
        RFDIR(NFR,1)=NR
        RFDIR(NFR,2)=FTAB(J,1)
        RFDIR(NFR,3)=IFOT
   60   CONTINUE
   70    IF(INA.EQ.0) GO TO 80
        INA=0
        N=NA
        IFOT=2
        GO TO 45
   80   IF(IDA.EQ.0) GO TO 110
        IDA=0
        N=DA
        IFOT=3
        GO TO 45
  110   IF(LIC.EQ.1) NR=INR
        RETURN
        END
```

```
      SUBROUTINE  DELTRL (ID,IABC)
C
C*****************************************************************************
C*******   THIS SUBROUTINE DELETES A RELATION AND PUTS THE AREA IN AVAL*******
C     EXPLANATION OF VARIBLES USED FOLLOWS---------------
C     ID       -----RELATION ID TO BE DELETED
C     IABC     -----USER ACCESS CODE  FOR USING THE PRIVILAGED ROUTINE
C*****************************************************************************
C*******           FINDING  RELID   IN   RTAB   USING   BSERCH    ************
      INTEGER RELID,FLDID,RFDIR,RTAB,FTAB,DNO,BUFF,ACCD,DA,CONT,DBA
      INTEGER AL,AV,FID
      DIMENSION KARD(40),FTAB( 40,4),RFDIR(50,3),RTAB(20,10),BUFF( 2,40)
     1,IREL(20),IRID(10),ACCD(2),MCFR(10,10),IFID(10),IUSR(40,40),DBA(2)
      DIMENSION  IABC(2),IV(40)
      COMMON KARD,NR,NF,NI,NA,FTAB,FLDID,RTAB,RELID,L,NDC,CONT,DNO,NOR
     1 ,NFR,BUFF,NRQ,IREL,IRID,IFLG,MCFR,RFDIR,ACCD ,DA,NAT,NII,NN,IFID
     2 ,IUSR,ITFLG,DBA
X     WRITE(6,1001)
      WRITE(2,1001)
 1001 FORMAT(30X,*ENTERING  SUBROUTINE----DELTRL  *)
C*******          I N I T I A L I A SA T I O N                  ***************
      AV=22081
      AL=19521
      CALL BSERCH (ID,RTAB,NR,IR)
C*********      TESTING  USER ACCES   CODES    FOR AUTHORISATOIN  *************
      IF(RTAB(IR,6).EQ.IABC(1) ) GO TO 10
X     WRITE(6,1) IABC(1)
      WRITE(2,1) IABC(1)
    1 FORMAT(10X,A2, *----ACCES CODE NOT MATCHEDBY USER *)
      RETURN
C********  TSETING  USER  ACCES  LEVEL FOR  USING  DELTRL CALL  ***************
   10 IF(IABC(2).LT.10) GO TO 15
X     WRITE(6,2) IABC(2)
      WRITE(2,2) IABC(2)
    2 FORMAT(10X,I2,*LEVEL GREATER THAN 10 UOSER NOT ALLOWED DELTRL*)
      IFLG=2
      RETURN
C*******   HAV*NG SATISIFIED USER AUTHORISATION PROCEEDING TO DELET   REL******
   15 JR=IR-1
      IF(( JR-(JR/6*6)).EQ.0) GO TO 17
      JO=0
      NDC=RTAB(JR,7 )
      IOR=NDC+NDC/10+1
      WRITE(6,1003) DNO,CONT,NI,NA,DA,NDC,IR,JR,JO,IOR
      CALL GETO(JR,IOR,JO)
      WRITE(6,1003) DNO,CONT,NI,NA,DA,NDC,IR,JR,JO,IOR
C********    BLANKING THE USER  AREA    FOR  INITIALISATION  *****************
   17 DNO=RTAB(IR,4)
      NDC=RTAB(IR,7 )
      NI=RTAB(IR,8)
      NA=RTAB(IR,9)
      DA=RTAB(IR,10)
      NAT=10*NA+30*DA
      NN=NI+NAT
      DO 16 NT=1,NN
      INOR=1
      IF(NT.LE.NI)BUFF(INOR,NT)=0
  16  IF(NT.GT.NI)BUFF(INOR,NT)=8224
C*****BLANKING  THE  DISK SPACE BY USING  IOPT    ROUTINE        **************
      NOR=1
      CONT=2
      WRITE(6,1003) DNO,CONT,NI,NA,DA,NDC,IR,NN
 1003 FORMAT(10I5)
      DO 30IK=1,NDC
      CALL IOPT
  30  CONTINUE
C******** REWINDING  THE PROPER DEVICE FOR FUTURE USE    ********************
      GO TO (61,62,63,64,66),DNO
  61  GO TO 36
  62  REWIND 13
      GO TO 36
  63  REWIND 14
      GO TO 36
  64  REWIND 15
      WRITE(2,1002)
      WRITE(6,1002)
 1002 FORMAT(10X,*DNO 4 HAS BEEN REWOUND*)
      GO TO 36
  66  REWIND 16
C****     UPDATING  RTAB  AND  RFDIR          *********************************
  36  RTAB(IR,1)=0
      RTAB(IR,2)=AV
      RTAB(IR,3)=AL
      WRITE(6,9009)(RTAB(IR,J),J=1,10)
 9009 FORMAT(I5,2A2,2I5,A2,4I5)
C********   FINDING  FIELD IN THE RELATION FOR UPDATING  FTAB ****************
      DO 40 M=1,NFR
      IF(RFDIR(M,1).EQ.ID)  GO TO 50
      GO TO 40
C******     DELETING  FTAB  ENTRY IF NOT REQUIRED BY ANY OTHER USER ***********
  50  RFDIR(M,1)=0
      DO 60 N=1,NFR
      IF(N.EQ.M) GO TO 60
      IF(RFDIR(N,2).EQ.RFDIR(M,2)) GO TO 40
  60  CONTINUE
      IFK=RFDIR(M,2)
      FTAB(IFK,1)=0
  40  CONTINUE
      RETURN
      END
```